М. С. Пасєка, Л. М. Гобир

Людино-машинна взаємодія

ЛАБОРАТОРНИЙ ПРАКТИКУМ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Івано-Франківський національний технічний університет нафти і газу

Кафедра інженерії програмного забезпечення

М. С. Пасєка, Л. М. Гобир

ЛЮДИНО-МАШИННА ВЗАЄМОДІЯ

ЛАБОРАТОРНИЙ ПРАКТИКУМ

Івано-Франківськ

2021

УДК 004.65
П –19

Рецензент:
Шекета В.І . д-р. техн. наук, проф., каф. інженерії програмного забезпечення ІФНТУНГ

Рекомендовано методичною радою університету
(протокол №3 від 29.04.2021 р.)

М.С.Пасєка, Л.М. Гобир
П- 19 Людино-машинна взаємоді:. лабораторний практикум для виконання лабораторних робіт. Івано-Франківськ: ІФНТУНГ, 2021. 82 с.

МВ 02070855-19060 - 2021

Лабораторний практикум для виконання лабораторних робіт з дисципліни «Людино-машинна взаємодія» складено відповідно до нового навчального плану та освітньо-кваліфікаційної програми для студентів за спеціальністю 121 «Інженерія програмного забезпечення».

Містить теоретичний матеріал, який стисло висвітлює найважливішу інформацію для знань студентів; виконання робіт супроводжується підказками, роз'ясненнями та ілюстраціями; перелік питань для контролю та атестації знань студентів. Призначений для студентів денної та заочної форм навчання.

МВ_____ -2021

© Пасєка М.С
© Гобир Л.М
© ІФНТУНГ, 2021

УДК 004.65
П –19

Рецензент:
Шекета В. І. д. т. н, проф., каф. інженерії програмного забезпечення ІФНТУНГ

Рекомендовано методичною радою університету
(протокол № від 2021р.)

М.С.Пасєка, Л.М. Гобир
П- 19 Людино-машинна взаємодія лабораторний практикум для виконання лабораторних робіт для студентів спеціальності 121 Інженерія програмного забезпечення. – Івано-Франківськ: ІФНТУНГ, 2021. – 82 с.

МВ _____ - 2021

Лабораторний практикум для виконання лабораторних робіт з дисципліни «Людино-машинна взаємодія» складено відповідно до нового навчального плану та освітньо-кваліфікаційної програми для студентів за спеціальністю 121 «Інженерія програмного забезпечення».

Містить теоретичний матеріал, який стисло висвітлює найважливішу інформацію для знань студентів; виконання робіт супроводжується підказками, роз'ясненнями та ілюстраціями; перелік питань для контролю та атестації знань студентів. Призначений для студентів денної та заочної форм навчання.

Зав. каф інженерії програмного забезпечення В.І.Шекета
Член експертно-ревізійної комісії В. В. Бандура

Нормоконтролер Г. Я. Томашівська

Провідний бібліотекар НТБ Г. М. Мацюк

МВ_____ -2021

УДК 004.65

© Пасєка М.С.,
© Гобир Л.М.,
© ІФНТУНГ, 2021

ЗМІСТ

ВСТУП

Людино-машинна взаємодія, вивчає способи, у які люди застосовують або не використовують обчислювальні артефакти, системи та інфраструктури. До того-ж, велику частину досліджень цієї області, спрямовано на поліпшення взаємодії людини з комп'ютером, за рахунок підвищення зручності використання комп'ютерних інтерфейсів. Усе частіше обговорюється — як, зручність і простота, співвідносяться з іншими соціальними і культурними цінностями, і як можна точно зрозуміти, коли вони потрібні і, коли вони не можуть бути, бажаною властивістю комп'ютерних інтерфейсів.

Велика частина досліджень з області взаємодії людини з комп'ютером, стосується:

- Методів розробки нових комп'ютерних інтерфейсів, що покращують конструкцію, для отримання бажаної властивості, такої як, наприклад, здатність до навчання або ефективність використання.
- Способів реалізації інтерфейсів, наприклад, за допомогою програмних бібліотек.
- Засобів оцінки та порівняння інтерфейсів щодо їх придатності, та інших бажаних властивостей.
- Способів вивчення використання комп'ютера людиною і їх соціально-культурних наслідків у більш широкому сенсі.
- Моделі та теорії використання комп'ютера людиною, а також концептуальних меж задля розробки комп'ютерних інтерфейсів, таких як, наприклад, когнітивні моделі користувачів, теорії діяльності або етно-методологічного пояснення використання комп'ютера людиною.
- Перспективи, як критично осмислювати цінності, які лежать в основі обчислювального проєктування, використання комп'ютера та науково-дослідну практику HCI.

Бачення того, що прагнуть досягти дослідники в області HCI, є різноманітними. Коли розробляють когнітивні плани на майбутнє, дослідники HCI, можуть спробувати вирівняти комп'ютерні інтерфейси з ментальною моделлю, яку люди використовують у своїй діяльності. За дотримання після-когнітивних перспектив, дослідники HCI, можуть спробувати вирівняти комп'ютерні інтерфейси з присутніми соціальними практиками або наявними соціально-культурними цінностями.

Дослідники з HCI, зацікавлені в розробленні нових методик проектування, експериментуванні з новими пристроями, прототипуванні нових програмних та апаратних систем, дослідженні свіжих парадигм взаємодії, та проектуванні моделей і теорій взаємодії.

Багато розроблених програм забезпечено яскравим і, на перший погляд, дуже вражаючим інтерфейсом у вигляді численних і різноманітних кнопок, піктограм, перемикачів тощо, які при усій зовнішній привабливості не вирішують головної задачі забезпечення ефективної взаємодії користувача з комп'ютером. З іншого боку, навіть дійсно корисні програми, але оснащені невдалим інтерфейсом, залишаються незатребуваними.

Важливість грамотного проектування і реалізації інтерфейсу користувача усвідомлена провідними фірмами-виробниками програмного забезпечення вже давно. Про це говорить той факт, що проектування інтерфейсу користувача розглядається ними як окремий процес у рамках життєвого циклу будь-якого програмного продукту.

Інтерфейс користувача – це сукупність інформаційної моделі проблемної області, засобів і способів взаємодії користувача з інформаційною моделлю, а також компонентів, які забезпечують формування інформаційної моделі в процесі роботи програмної системи.

Під інформаційною моделлю розуміється умовне представлення проблемної області, що формується за допомогою комп'ютерних (візуальних і звукових) об'єктів, що відображають склад і взаємодію реальних компонентів проблемної області. Засоби і способи взаємодії з інформаційною моделлю визначаються складом апаратного і програмного забезпечення, наявного у розпорядженні користувача, і від характеру вирішуваної задачі.

Поширеною є ситуація, коли програми, рівноцінні за призначенням і функціональним можливостям, виявляються зовсім різними по організації взаємодії з користувачем. При цьому не обов'язково інтерфейс якоїсь з програм буде гірше, він просто буде іншим.

Ефективність роботи користувача визначається не лише функціональними можливостями наявних в його розпорядженні апаратних і програмних засобів, але і доступністю для користувача цих можливостей. У свою чергу, повнота використання потенційних можливостей наявних ресурсів залежить від якості інтерфейсу користувача.

Якість інтерфейсу користувача є самостійною характеристикою програмного продукту, порівняно за значимістю з такими його показниками, як надійність і ефективність використання обчислювальних ресурсів. Розробник програмного продукту повинен знати, що таке хороший інтерфейс і як його спроектувати. Цим питанням і присвячена дисципліна «Людино-машинна взаємодія».

ЛАБОРАТОРНА РОБОТА № 1

Основи проектування та дизайн інтерфейсу користувача

Мета роботи: ознайомитися з основними принципами проектування інтерфейсу користувача та дослідити найпоширеніші елементи з яких складаються сучасні людино-машинні інтерфейси.

Теоретичні відомості

1.1 Основні принципи дизайну інтерфейсу користувача

Взаємодія між користувачем і комп’ютером (HCI - Human-Computer Interaction) відбувається в інтерфейсі. Основною метою HCI є покращення взаємодії між користувачем і комп’ютером, шляхом підвищення рівня корисності та сприйнятливості останнього до потреб користувачів.

Найчастіше ефективність використання всіх функцій системи та ефективність роботи самої системи визначається рівнем побудови її інтерфейсу.

Природність інтерфейсу

Природний інтерфейс — це такий інтерфейс, що не потребує від користувача істотної зміни звичних для нього способів розв’язання задачі. Це, зокрема, означає, що повідомлення і результати, які отримуються за допомогою програмного продукту, не повинні вимагати додаткових пояснень. Доцільно також зберігати систему позначень і термінологію, які використовуються в даній предметній галузі.

Використання знайомих користувачу понять і образів (метафор) забезпечує інтуїтивно зрозумілий інтерфейс при виконанні завдань.

Метафори є свого роду «містком», що зв'язує образи реального світу з тими діями і об'єктами, якими доводиться маніпулювати користувачу при його роботі на комп'ютері; вони забезпечують «пізнавання», а не «згадку». Користувачі запам'ятовують дію, пов'язану із знайомим об'єктом, легше, ніж коли б їм необхідно було б запам'ятати ім'я команди, пов'язаної з цією дією.

Узгодженість інтерфейсу

Узгодженість інтерфейсу означає використання однакових або дуже схожих метафор і способів взаємодії з користувачем та порядку роботи в різних додатках.

Узгодженість надає можливість користувачам переносити наявні знання на нові завдання, освоювати нові аспекти швидше, і завдяки цьому фокусувати увагу на задачу, що вирішується, а не витрачати час на з'ясування відмінностей у використовуванні тих або інших елементів управління, команд і т. д. Забезпечуючи спадкоємність одержаних раніше знань і навичок, узгодженість робить інтерфейс зрозумілим і передбачуваним.

Узгодженість важлива для всіх аспектів інтерфейсу, включаючи імена команд, візуальне представлення інформації і поведінку інтерактивних елементів.

Узгодженість в межах продукту

Одна і та ж команда повинна виконувати одні і ті ж функції, де б вона не зустрілася, причому однаково.

Узгодженість в межах робочого середовища

Підтримуючи узгодженість з інтерфейсом, що надається операційною системою (наприклад, ОС Windows), програмний продукт може «спиратися» на ті знання і навички користувача, які він одержав раніше при роботі з іншими програмними продуктами.

Узгодженість у використанні метафор

Якщо поведінка деякого програмного об'єкту виходить за рамки того, що звичайно мається на увазі під відповідною йому метафорою, у користувача можуть виникнути труднощі при роботі з таким об'єктом. Наприклад, якщо для програмного об'єкту *Кошик* визначити операцію *Запуск,* то для з'ясування його призначення користувачу, швидше за все, знадобиться стороння допомога.

Дружність інтерфейсу

Нові користувачі вивчають особливості роботи з новим програмним продуктом методом спроб і помилок. Ефективний інтерфейс повинен брати до уваги такий підхід: на кожному етапі роботи він повинен надавати користувачу тільки відповідний набір дій і попереджати користувачів про ті ситуації, де вони можуть нашкодити системі або даним. Ще краще, якщо у користувача існує можливість відмінити або виправити виконані дії.

Навіть за наявності добре спроектованого інтерфейсу користувачі можуть робити ті або інші помилки. Ці помилки можуть бути як «фізичного» типу (випадковий вибір неправильної команди або даних), так і «логічного» (ухвалення неправильного рішення при виборі команди або даних). Ефективний інтерфейс повинен запобігати ситуаціям, які, ймовірно закінчаться помилками. Він також повинен уміти адаптуватися до потенційних помилок користувача і полегшувати йому процес усунення наслідків таких помилок.

Принцип «зворотного зв'язку»

Програмний продукт повинен забезпечувати зворотній зв'язок для дій користувача. Кожна дія користувача повинна одержувати візуальне, а іноді і звукове підтвердження того, що програмне забезпечення сприйняло введену команду; при цьому вид реакції, по можливості, повинен враховувати природу виконаної дії.

Зворотній зв'язок ефективний в тому випадку, якщо він реалізується своєчасно, тобто якомога ближче до точки останньої взаємодії користувача з системою. Коли комп'ютер обробляє завдання, що поступило, корисно надати користувачу інформацію щодо стану процесу, а також можливість перервати цей процес у разі потреби. Ніщо так не бентежить не дуже досвідченого користувача, як заблокований екран, який ніяк не реагує на його дії. Типовий користувач здатний витерпіти тільки декілька секунд очікування відповіді реакції від комп’ютера.

Простота інтерфейсу

Інтерфейс повинен бути простим. При цьому мається на увазі не спрощення, а забезпечення легкості в його вивченні і в користуванні. Крім того, він повинен надавати доступ до всього переліку функціональних можливостей, передбачених програмним продуктом. Реалізація доступу до широких функціональних можливостей і забезпечення простоти роботи суперечать одна одній. Розробка ефективного інтерфейсу покликана збалансувати ці цілі.

Гнучкість інтерфейсу

Гнучкість інтерфейсу — це його здатність враховувати рівень підготовки і продуктивність праці користувача. Властивість гнучкості припускає можливість зміни структури діалогу та вхідних даних. Концепція гнучкого (адаптивного) інтерфейсу в даний час є однією з основних областей дослідження взаємодії людини і комп'ютера. Основна проблема полягає не в тому, як організувати зміни в діалозі, а в тому, які ознаки потрібно використовувати для визначення необхідності внесення змін і їх суті.

Естетична привабливість

Проектування візуальних компонентів є найважливішою складовою частиною розробки програмного інтерфейсу. Коректне візуальне представлення використовуваних об'єктів забезпечує передачу досить важливої додаткової інформації про поведінку і взаємодію різних об'єктів. У той же час слід пам'ятати, що кожен візуальний елемент, який з'являється на екрані, потенційно вимагає уваги користувача, яка, як відомо, не безмежна. Необхідно забезпечити формування на екрані такого середовища, яке не тільки сприяло б розумінню користувачем представленої інформації, але і дозволяло б зосередитися на найважливіших її аспектах.

Золотий перетин

Золотий перетин – це сама комфортна для ока пропорція, форма, в основі побудови якої лежить поєднання симетрії і золотого перетину, сприяє найкращому зоровому сприйняттю і появі відчуття краси і гармонії.

У математиці пропорцією називають рівність двох відношень: $a/b = c/d$.

Відрізок прямої AB можна розділити точкою C на дві частини наступними способами:

- на дві рівні частини AB/AC = AB/BC;
- на дві нерівні частини в будь-якому відношенні (такі частини пропорції не утворюють);
- таким чином, коли AB/BC = BC/AC.

Останнє і є золотим перетином або розподілом відрізка в крайньому і середньому відношенні.

Золотий перетин – це таке пропорційне ділення відрізка на нерівні частини, при якому весь відрізок так відноситься до більшої частини, як сама велика частина відноситься до меншої; або іншими словами, менший відрізок так відноситься до більшого, як більший до всього відрізка.

$$a/b = b/c \text{ або } c/b = b/a.$$

Відрізки золотої пропорції виражаються нескінченним ірраціональним дробом 0,618 ..., якщо *c* прийняти за одиницю, *a* = 0,382. Відношення ж відрізків *a* і *b* складає 1,618.

Прямокутник з таким відношенням сторін стали називати золотим прямокутником. Він також володіє цікавими властивостями. Якщо від нього відрізати квадрат, то залишиться знову золотий прямокутник. Цей процес можна продовжувати до нескінченності. А якщо провести діагональ першого і другого прямокутника, то точка їх перетину належатиме всім одержуваним золотим прямокутниках.

Є і золотий трикутник (рівнобедрений трикутник, у якого відношення довжини бічної сторони до довжини основи дорівнює 1,618), і золотий паралелограм (прямокутний паралелепіпед з ребрами, що мають довжини 1,618, 1 і 0,618).

Золотий перетин не є штучним явищем. Воно дуже широко поширене в природі: золотий перетин можна знайти в пропорціях тіл багатьох рослин і тварин, а також морських раковин і пташиних яєць. Але найбільш вражаючий приклад "застосування" природою принципу золотого перетину є людське тіло. Воно і його частини (обличчя, руки, кисті рук і т. д.) наскрізь пронизані пропорцією 1,618.

Принцип золотого перетину був відкритий людьми ще в глибоку давнину. Знамениті єгипетські піраміди в Гізі, наприклад, засновані на пропорціях золотого перетину. Більш молоді мексиканські піраміди і античний храм Парфенон також містять в собі пропорцію 1,618.

З розвитком дизайну й технічної естетики чинність закону золотого перетину поширилася на конструювання машин, меблів і т. д. Проектування комп'ютерних інтерфейсів не є виключенням. Форми діалогових вікон і елементів управління, сторони яких відносяться як 1,618, дуже привабливі для користувачів.

Гаманець Міллера

Цей принцип названий так на честь вченого-психолога Г. А. Міллера, який досліджував короткотривалу пам'ять, перевіряючи висновки, зроблені раніше його колегою, Г. Еббінгаузом. Еббінгауз намагався з'ясувати, скільки інформації може запам'ятати людина без будь-яких спеціальних мнемонічних прийомів. Виявилося, що ємність пам'яті обмежена сімома цифрами, сімома літерами або назвами семи предметів. Це "магічне число" сім, що служить свого роду міркою пам'яті, і було перевірено Міллером, який показав, що пам'ять дійсно в середньому не може зберігати більше семи елементів. В залежності від складності елементів це число може коливатися в межах від п'яти до дев'яти.

Якщо необхідно протягом короткого часу зберегти інформацію, що включає більше семи елементів, мозок майже несвідомо групує цю інформацію таким чином, щоб число елементів, які необхідно запам'ятовувати не перевищувало гранично допустимого. Наприклад, номер банківського рахунку 30637402710, що складається з одинадцяти елементів, буде, швидше за все, запам'ятовуватися як 30 63 740 27, тобто як п'ять числових елементів, або вісім слів (тридцять, шістдесят, три, сімсот, сорок, двадцять, сім, десять).

Застосовуючи принцип гаманця Міллера в дизайні інтерфейсів, слід групувати елементи в програмі (кнопки на панелях інструментів, пункти меню, закладки, опції на цих закладках і т. д.) з урахуванням цього правила, тобто не більше семи в групі, в крайньому випадку – дев'ять. Погляньте, наприклад, на головне вікно програми-словника ABBYY Lingvo: чотирнадцять кнопок на верхній панелі, між якими немає жодного роздільника, сприймаються набагато гірше, ніж кнопки на панелі внизу, які розділені на групи.

Отже, принцип гаманця Міллера каже про сім плюс-мінус два елементи. Але якщо поглянути на програми, інтерфейс яких удосконалювався роками (той же Microsoft Word), то можна помітити, що число об'єктів (пунктів меню, кнопок на панелях інструментів) в групах доходить до шести-семи досить рідко, а в основному елементи згруповані по три-чотири об'єкта. Такі невеликі групи об'єктів найбільш добре сприймаються користувачем, який уже трохи втомлений складними інтерфейсами сучасних програм. При проектуванні інтерфейсів програм, верхню межу гаманця Міллера (сім-дев'ять елементів) потрібно застосовувати дуже обережно, намагаючись обходитися групами, що містять максимум п'ять об'єктів.

Принцип угруповання

Згідно з цим правилом, екран програми повинен бути розбитий на ясно обкреслені блоки елементів, які можуть містити заголовки для кожного блоку. При цьому групування, повинно бути осмисленим: як розташування елементів в групах, так і розташування самих груп одна відносно одної повинні бути продумані.

Бритва Оккама або KISS

Філософський принцип, що носить назву "Бритва Оккама", говорить: "Не множити сутності без потреби". Або, як кажуть американці, KISS ("Keep It Simple, Stupid" - "Не ускладнюй, телепень").

Мовою інтерфейсів це означає, що:

- будь-яка задача повинна вирішуватися мінімальним числом дій;
- логіка цих дій повинна бути очевидною для користувача;
- рухи курсору і навіть очей користувача повинні бути оптимізовані.

Розумне запозичення

Запозичення широко поширених прийомів дизайну інтерфейсів і вдалих винаходів авторів конкуруючих програм дозволяє різко скоротити час навчання і підвищити комфорт користувача. При роботі він використовуватиме вже придбані навички. Запозичення елементів з чужих інтерфейсів не є чимось ганебним.

1.2 Етапи створення інтерактивних прототипів

Етап І. Передпроектний аналіз

Роботи з проектування інтерфейсу починаються з передпроектного аналізу. На робочій зустрічі з клієнтом описується бачення проекту (vision), в якому розповідається про його суть і цілі, а також перераховується передбачувана функціональність системи у вигляді коротких сценаріїв взаємодії. На додаток

до цього проводиться аналіз потреб і контексту роботи цільової аудиторії, яка описується у вигляді ключових персонажів. Також складається первинна карта системи, яка показує приблизну структуру майбутньої системи. На написання і затвердження цих базових документів зазвичай йде близько 3 днів, після чого плануються інші роботи і дається точна оцінка термінів і вартості їх виконання. Тому зручніше вести передпроектний аналіз за окремим договором – складно отримати точну оцінку двомісячної роботи без попереднього дослідження.

Етап II. Збір вимог

На наступному етапі готується докладний перелік функціональності (user stories). Він дозволяє врахувати всі функціональні вимоги і краще зрозуміти особливості майбутньої системи. На його основі робиться висновок, які з функцій вимагають цілого процесу, які окремого вікна, а яким буде досить простої кнопки.. Після цього рисуються діаграми переходів між вікнами. Вони об'єднують вікна системи в рамках конкретних процесів. Тепер ми знаємо, як користувачі будуть працювати з продуктом в цілому і як саме виконувати конкретні завдання. Етап триває близько 4 днів.

Етап III. Проектування інтерфейсу

Третій етап – найважливіший. Тут необхідно створити структурні схеми сторінок (wireframes), які показують, яка інформація і елементи управління повинні розташовуватися на сторінках системи. Це ще не дизайн, але вже основа для нього – wireframes є технічним завданням для дизайнера. Спілкування з клієнтом на цьому етапі досить щільне – уточнення питань і затвердження креслень йде по кілька разів на день. Але і результатів вистачає – в залежності від складності проекту виходить від кількох десятків до пари сотень схем сторінок. Тривалість етапу – від одного до декількох тижнів.

Етап IV. Дизайн інтерфейсу

Завершальним етапом стає візуальний дизайн інтерфейсу. Спершу на основі пари ключових сторінок відпрацьовується креативна концепція. Після того як загальна стилістика схвалена клієнтом, малюються дизайн-макети ключових сторінок системи. На цьому етапі продукт отримує зовнішній вигляд. Для проектів, які планують активно розвиватися, готується довідник щодо стилю інтерфейсу (style guide). Він описує принципи візуального оформлення продукту і дозволяє зберегти його цілісність у процесі доопрацювань. Роботи на цьому етапі тривають 1-2 тижні.

Етап V. Підготовка специфікації

При необхідності готується попереднє технічне завдання на розробку системи. Воно об'єднує в собі отримані раніше документи, розширює і перераховує додаткові вимоги до системи – функціональні, архітектурні, експлуатаційні. За бажанням клієнта можуть бути складені детальні сценарії взаємодії, які крок за кроком описують процес роботи користувача з системою.

Фінальний етап. Приймання

Приймання роботи клієнтом може проходити одним великим пакетом зауважень або розбиватися на декілька більш дрібних етапів. Терміни, в які

зауваження повинні бути виставлені, оцінені і виправлені обумовлюються в договорі.

У залежності від стадії проектування і цілей прототипу, необхідно дотримуватися трьох основних критеріїв:

Візуальна точність (почерк ↔ дизайн)

Можливість побачити і відчути – найважливіша характеристика прототипу, якщо спроектувати його невірно – вся робота може виявитися марною. З візуальної точки зору прототипи не повинні бути дуже красивими, але повинні бути пропорційними. Наприклад, якщо ліва навігаційна область повинна зайняти одну п'яту екрану на 1024 пікселя, вона не повинна бути точно 204 пікселя в ширину зображена в прототипі. Безпосередньо в процесі проектування дизайну необхідно збільшити візуальну точність, вводячи елементи стилю, кольору, айдентики і графіки.

Функціональна точність (статичний ↔ інтерактивний)

Прототип тільки показує як кінцевий продукт буде працювати (статичний прототип) або він є функціональний (інтерактивний прототип) і може взаємодіяти з користувачем? Це питання не настільки важливе для замовника, але помітно додає ясності на наступному етапі розробки, збільшує функціональну точність і дозволяє використовувати прототип для перевірки зручності роботи із реальною системою.

Точність контенту (lorem ipsum ↔ реальний контент)

Інша важлива складова, яка часто відволікає користувачів – контент, відображений у прототипі. Абстрактні рядки і фіктивний текст (наприклад
«lorem ipsum») корисні, щоб уникнути деяких складнощів на ранніх етапах проектування. Але оскільки прототип постійно вдосконалюється, виникає необхідність замінити фіктивний текст реальним контентом, щоб отримати уявлення про те, як це буде позначатися на розробці дизайну.

Спектр проектування

Низька точність. Найшвидший спосіб почати проектувати є також і самим простим: почніть з паперу та ручки. Малювання ескізів на папері – підхід з низькою точністю, який може зробити практично будь-яка людина. Ніякі спеціальні інструментальні засоби або досвід не потрібні. Найчастіше використовується під час ранніх стадій циклу дизайну, малювання ескізів – швидкий спосіб створити грубі макети зразків дизайну і приблизну концепцію продукту, а також отримати зворотній зв'язок від клієнтів.

Паперове макетування ідеальне під час мозкового штурму і може бути зроблене на самоті із блокнотом або під час наради на дошці маркерами. Паперові прототипи статичні і зазвичай відображають тільки приблизний вигляд продукту – це змушує клієнта зосередитися на тому, як вони будуть використовувати систему замість того, щоб думати на що вона буде схожа.

Проектування низької точності – швидке проектування, воно не вимагає особливих навичок, але дозволяє вносити легко і швидко зміни в проект.

Середня точність. На цьому етапі використовується ПЗ, таке як Visio і Omnigraffle, для моделювання і збільшення точності макетів. Каркаси, завдання і сценарії, створені за допомогою даних інструментів, займають більше часу і

сил, але в результаті ми отримуємо акуратний і більш досконалий прототип. На цьому етапі візуальні елементи, такі як айдентіка, кольори вже можуть використовуватися, але проектувальники часто уникають їх, зосереджуючись натомість на демонстрації поведінки програми в цілому. Інтерактивність може бути показана шляхом, створення посилань, але функціональну точність тут краще залишити середньої якості. Такі прототипи найкраще підходять для того, щоб зрозуміти, чи задоволені клієнтські потреби і оптимально розбудувати макет з точки зору юзабіліті.

Є дві причини, чому необхідно навмисно робити прототип середньої точності не схожим на прототип середньої точності:

- перш за все, роблять так, щоб прототип виглядав як макет з низькою точністю для того, щоб клієнти розглядали його як нарис або макет в стадії розробки, а не готовий виріб;
- по-друге, надаючи прототипу високу візуальну точність (наприклад, у вигляді гарного макета, зробленого в Photoshop), Ви змушуєте клієнта зосереджуватися на оцінці якості візуального дизайну – кольорах, шрифтах, логотипах і зображеннях.

Швидкість створення прототипу середньої точності досягається використанням шаблонів, трафаретів і типових графічних елементів.

Висока точність. Високоточні прототипи є найбільш реалістичними і часто помилково приймаються за кінцевий продукт. Кілька років тому, єдиний спосіб створити високоточні прототипи полягав у тому, що фактично створювався запрограмований прототип, а для цього над створенням прототипу довелося б працювати вже двом фахівцям – програмісту і проектувальнику. Зараз, на щастя, засоби прикладного моделювання дозволяють всім користувачам (навіть не сильно підкованих технічно) перетягувати і опускати графічні фрагменти інтерфейсу, щоб створити високоточні прототипи, які моделюють функціональні можливості кінцевого продукту.

Ці прототипи необхідні, коли потрібна висока візуальна і функціональна точність. Наприклад, при введенні нової технології. Більшість цих прототипів не може бути перетворено в придатний для використання код, але вони служать чудовою довідковою інформацією для розробників, а також для проведення перевірки зручності використання або навчання користувачів.

Високоточне проектування відносно швидке, при розумному рівні інтерактивності і точності. Крім того, деякі з цих інструментальних засобів полегшують встановлення зворотного зв'язку з користувачами, прискорюючи процес проектування і розробки. Не дивлячись на те, що в даному випадку не потрібно вивчати нову мову програмування, такі інструментальні засоби все рівно вимагають певного рівня освоєння.

Завдання на лабораторну роботу

1. Ознайомитися з теоретичними відомостями по роботі.
2. Знайти опис найпоширеніших елементів, які використовуються для формування інтерфейсів взаємодії людини з програмною системою.
3. Зробити висновки по роботі.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що ви розумієте під поняттям «принципи проектування інтерфейсу»?
2. Які принципи проектування інтерфейсу вам відомі?
3. В чому полягає структурний принцип?
4. Розкажіть про принцип простоти.
5. Що ви можете сказати про принцип видимості?
6. В чому полягає принцип зворотного зв’язку?
7. Охарактеризуйте принцип толерантності.
8. Що ви знаєте про принцип повторного використання?

ЛАБОРАТОРНА РОБОТА № 2

Проектування інтерфейсу користувача засобами Axure RP

Мета роботи: ознайомлення з програмою Axure RP, як з інструментом, призначеним для створення програмних прототипів і специфікацій.

Теоретичні відомості

Axure RP (укр. Ак-шур Ер-Пі) - програмне забезпечення для створення прототипів і специфікацій веб-сайтів і додатків.

На рисунку 2.1 представлений вид програми.

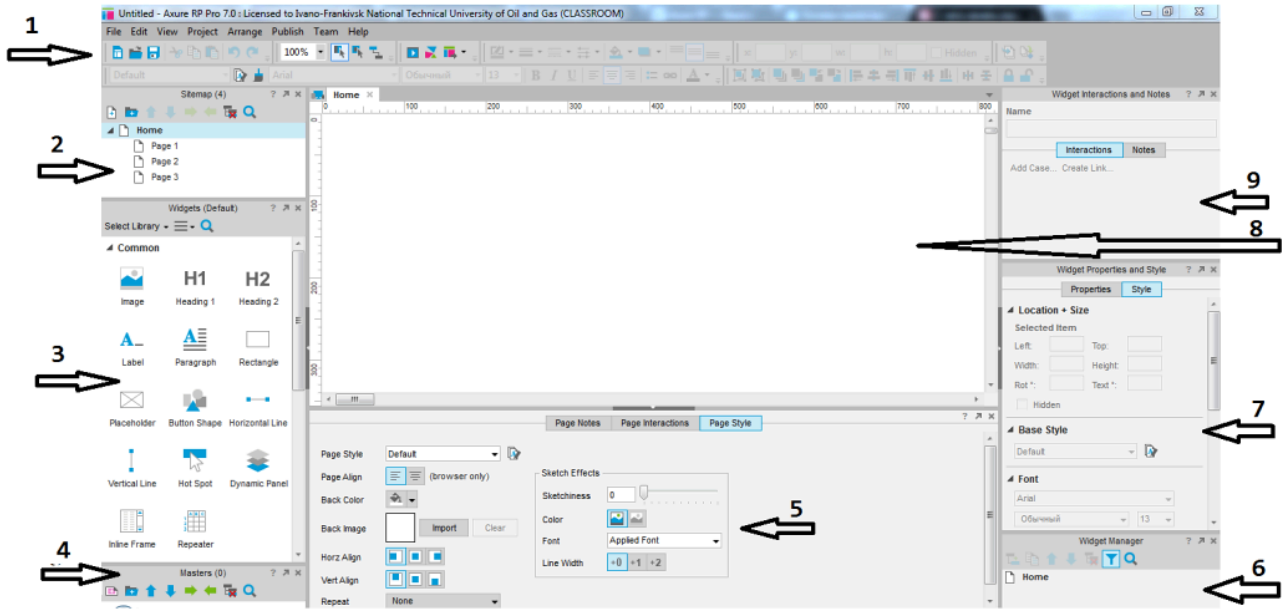


Рисунок 2.1 – Вид головного вікна Axure RP

Вікно середовища Axure RP розділено на 9 вікон. Верхнє вікно є основним вікном середовища. Воно містить:

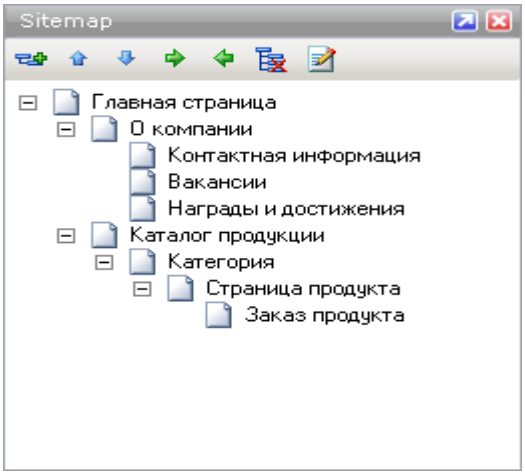
- основне меню Axure RP (рис. 2.1 елемент №1);
- карта сайту **Sitemap** (рис. 2.1 елемент №2);
- панель віджетів **Widgets** (рис. 2.1 елемент №3);
- панель майстерів **Masters** (рис. 2.1 елемент №4);
- панель властивостей Робочої області (рис. 2.1 елемент №5);
- панель карти віджетів **Widget Manage** (рис. 2.1 елемент №6);
- панель властивості віджетів Widget **Properties and Style** (рис. 2.1 елемент №7);
- робоча область (рис. 2.1 елемент №8);
- панель відджати взаємодії **Widget interactions and Notes** (рис. 2.1 елемент №9).

Оперативна панель дозволяє одним натисканням по відповідній кнопці виконувати

такі дії, як відкриття, збереження та компіляція проектів. Палітра компонентів містить великий набір компонентів, які ви можете вставляти у свої форми (до компонентів відносяться текстові мітки, вікна списку, кнопки і т.д.). Для зручності всі компоненти розділені на групи.

1. Карта сайту (Sitemap)

У цій області можна скласти структурну схему сайту, використовуючи багаторівневу ієрархію. Наприклад, структура невеликого проекту може виглядати так:



Кнопки в панелі призначені для швидкого доступу до основних функцій:

Створює вкладену сторінку (Child page)

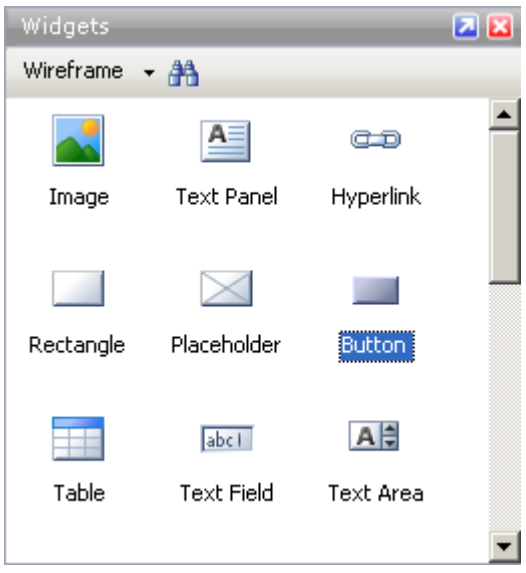
Переміщують обрану сторінку вгору або вниз. Працюють тільки зі сторінками одного рівня і не витягують їх за межі батьківського елементу. Якщо треба виділити і перемістити вгору або вниз відразу кілька сторінок - можна використовувати Shift.

Змінюють рівень вкладеності сторінок. Стрілка вліво виносить вибрані сторінки на рівень вище, стрілка вправо підпорядковує сторінку батьківського елементу, розташованому над нею.

Видаляє сторінку. У тому випадку, якщо батьківський елемент містить вкладені елементи, вони будуть видалені разом з ним.

Дозволяє перейти до редагування сторінки, т. Е. Робить те ж саме, що і подвійний клік по сторінці.

2. Панель віджетів



Панель містить набір інтерфейсних елементів, які постійно використовуються при роботі над проектом. Стандартні бібліотеки містять тільки найнеобхідніше - прямокутники, текстові панелі, плейсхолдери, кнопки, елементи форм і т.д. В область можна довантажувати або всі елементи з усіх бібліотек одночасно (для цього потрібно вибрати All libraries),

або тільки ту бібліотеку елементів, яка потрібна зараз. До речі, бібліотеки елементів можна створювати самостійно, але про це мова піде трохи нижче. Щоб помістити елемент на сторінку, використовується метод Drag and Drop (елемент необхідно виділити і, не відпускаючи кнопку миші, перетягнути в робочу область). Основна бібліотека, яку варто використовувати на стадії освоєння програми, називається Wireframe. Її елементи ми і будемо використовувати.


Image

Заглушка для зображення. Стандартний розмір - 50 × 50px.


Text Panel

Текстове поле (100 × 16px). За замовчуванням використовується Arial, 10, чорний колір.


Hyperlink

Гіперпосилання (100 × 16px). За замовчуванням використовується Arial, 10, синій колір + підкреслення.


Rectangle

Прямокутник 180 × 80px з білою заливкою і чорною рамкою в 1px.


Placeholder

Плейсхолдер, призначений, наприклад, для забивання баннерного місця. 180 × 80px, рамка і діагоналі - чорні, 1px.


Button

Кнопка (100 × 25px).


Table

Таблиця. За умовчанням створюється таблиця 3 × 3. Користуватися елементом не дуже зручно, т. К. Ширина задається тільки всій таблиці і не може здаватися певним стовпцях. Якщо ж починати руками зміщувати кордону стовпців усередині таблиці - ширина збільшується і зменшується за рахунок загальної ширини таблиці, а не інших стовпців.


Text Field

Поле для введення тексту (один рядок).


Text Area

Область для введення тексту (будь-яку кількість рядків і стовпців).


Droplist

Випадаючий список.


List Box

Багаторядковий список.

Чекбокс.

Радіокнопка.

Горизонтальна лінія.

Вертикальна лінія.

Кнопка з округленими кутами. Може з кнопки легко перетворитися на прямокутник або квадрат. Радіус скруглення можна задавати вручну, але тільки «на око». Поле для введення точного радіуса розробники полінувалися зробити.

Область накладення для зображень.

Кадр, куди може подружаться інформація з інших сторінок прототипу.

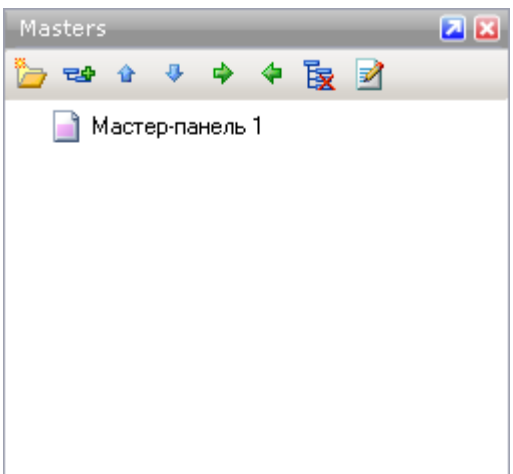
Динамічна панель. Може використовуватися, наприклад, для проставлення активності пунктів меню на певних сторінках. У цій статті питання інтерактивності прототипу розглядатися будуть дуже поверхнево.

Вертикальне багаторівневе меню, що випадає.

Горизонтальне багаторівневе меню, що випадає.

Список, що розкривається.

3. Панель майстрів (Masters)



Панель майстрів (Masters) У цій панелі розміщуються елементи, які багаторазово використовуються на сторінках сайту. Наприклад, щоб при дрібному зміні в футере не переробляти його на всіх розроблених сторінках, досить зробити його майстер-панеллю і

довантажувати на інші сторінки. Відповідно, щоб внести зміни на всіх сторінках, потрібно один раз підправити майстер-панель.

Майстер-панелі теж можуть бути багаторівневі (наприклад, футер може містити дочірні елементи «контакти» і «лічильники»).

За замовчуванням майстер-панелей немає. Щоб завести її і додати на всі сторінки, треба виконати послідовність:

Тиснемо кнопку AddMaster () → Два рази клікаємо по створеній майстер-панелі (в робочій області відкривається вкладка, на якій буде редагуватися вміст майстер-панелі) → Редагуємо майстер-панель, додаємо на неї необхідні елементи → Правою кнопкою миші по назві майстер-панелі викликаємо контекстне меню → Тиснемо «Add To Pages» → Вибираємо необхідні сторінки (якщо майстер-панель повинна виводитися на всіх сторінках, можна скористатися кнопкою «Check All») → Вибираємо позиціонування панелі (у тому випадку, якщо буде вибрано Place in background, майстер -панель збереже те ж саме розташування, в якому вона виконана. Якщо вибрати Specify Location і задати лівий і верхній відступи, майстер-панель займе зазначене положення на сторінках) → Тиснемо «ОК» і насолоджуємося розміщеної майстер-панеллю.

Якщо майстер-панель стала не потрібна і її хочеться видалити - спочатку треба скасувати її розміщення на сторінках прототипу, а тільки потім видаляти. Інакше вона буде чинити опір і лаятися.

4. Робоча область

Власне, в цій області і відбувається все найцікавіше - редагуються елементи та їх оформлення, оформлюються функціональні блоки і так далі.

Я думаю, найбільш наочним способом продемонструвати, як використовувати робочу область і працювати з елементами, буде покроковий опис створення головної сторінки чого-небудь.

Скажу відразу, що до кінця прототип розроблений не буде - зате тим, хто зацікавився Axure, буде надана можливість завантажити недопрацьований проект і доробити його самостійно.

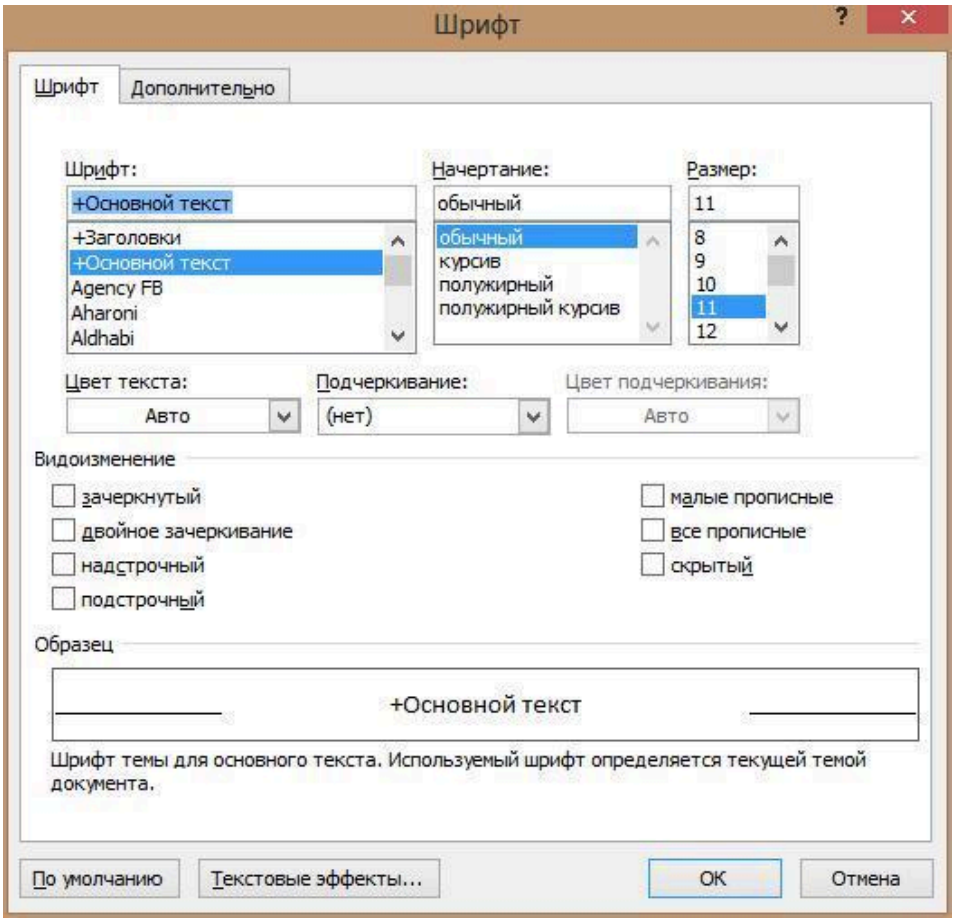
Майстер-панелі використовуватися не будуть, т. К. Ми наперед упевнені в тому, що розробляється тільки одна сторінка.

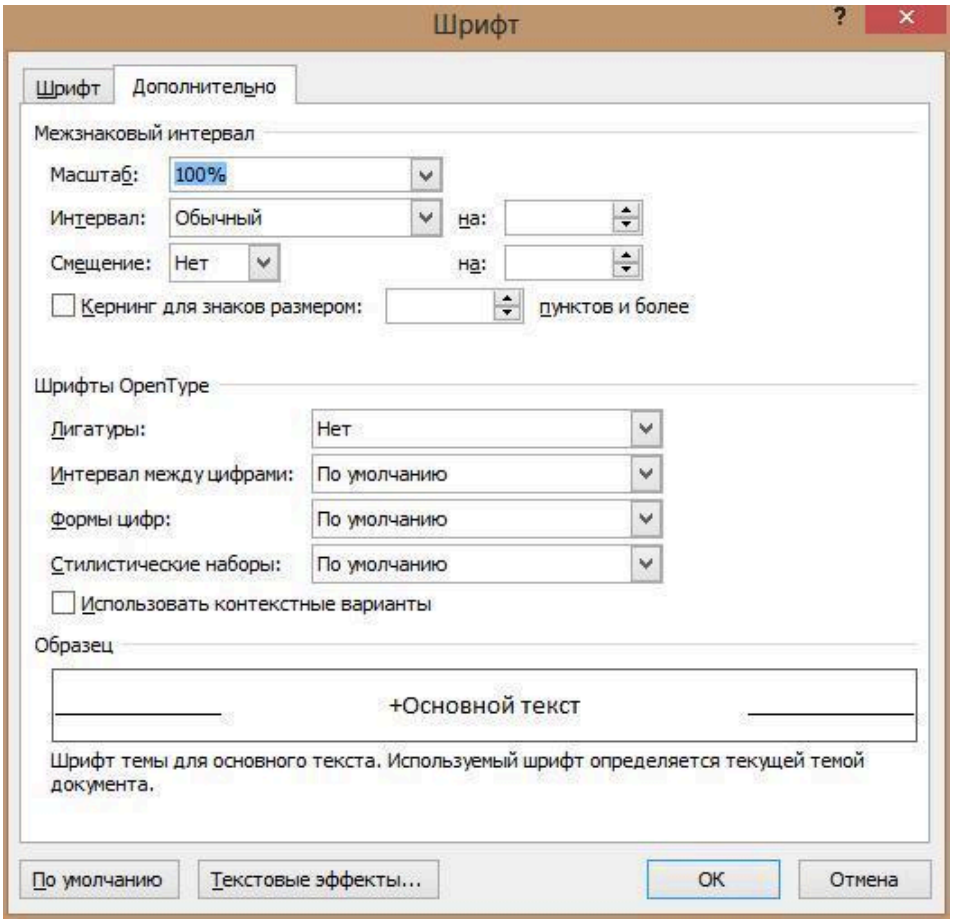
Завдання на лабораторну роботу

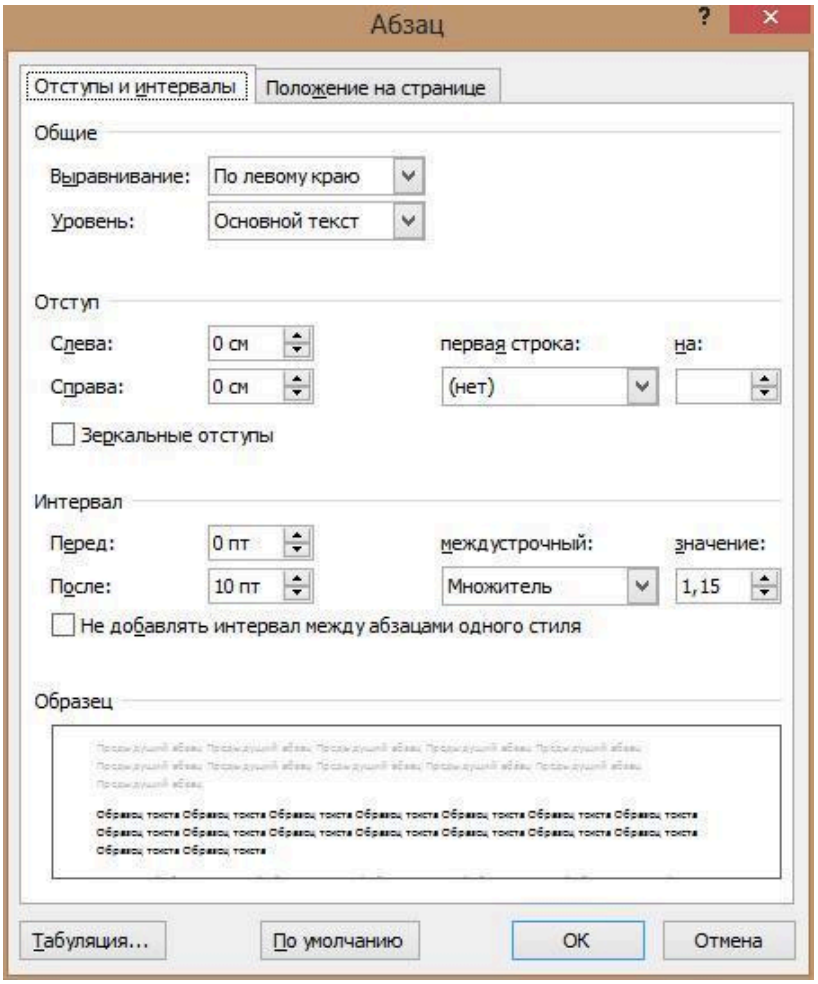
1. Ознайомитися з теоретичними відомостями по роботі.
2. Спроектувати графічний інтерфейс згідно свого варіанту вказаного в кінці лабораторної роботи.

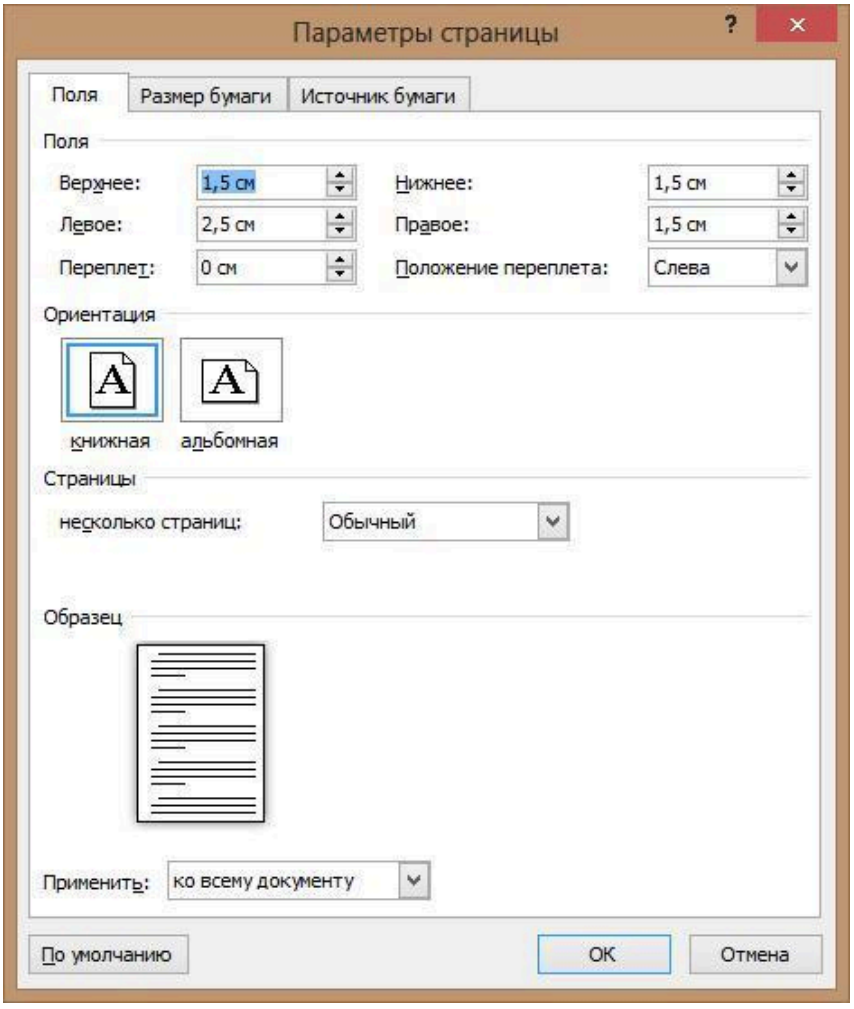
3. Зробити висновки по роботі.

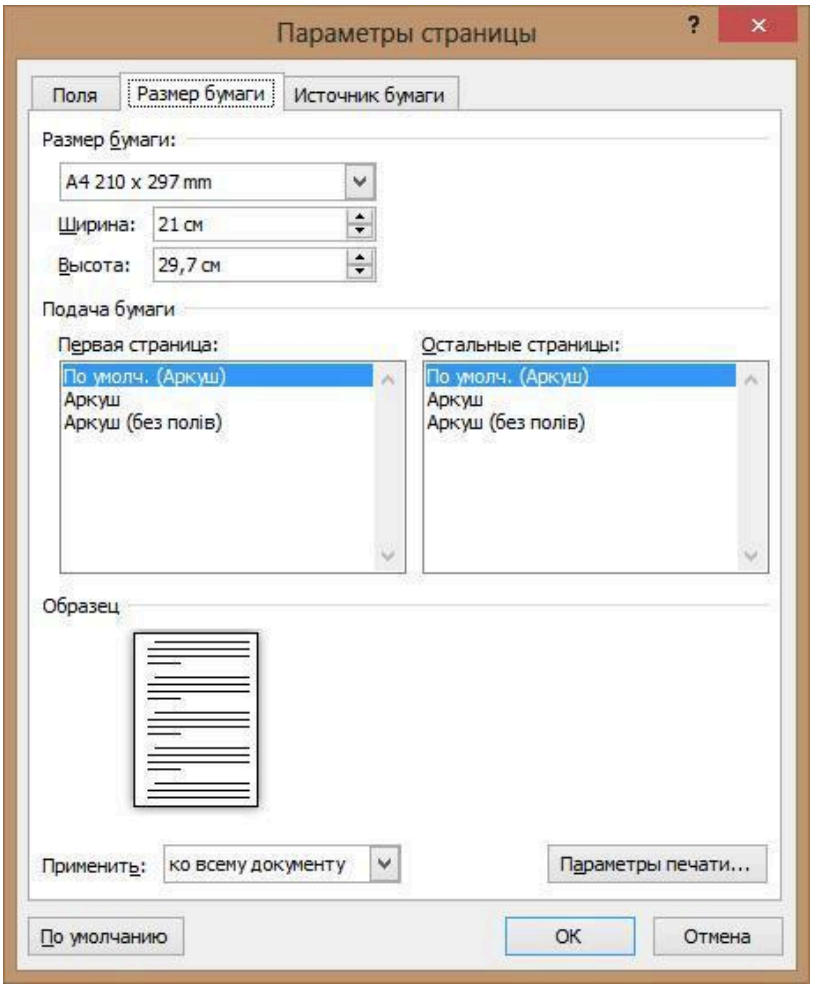
Варіанти завдань

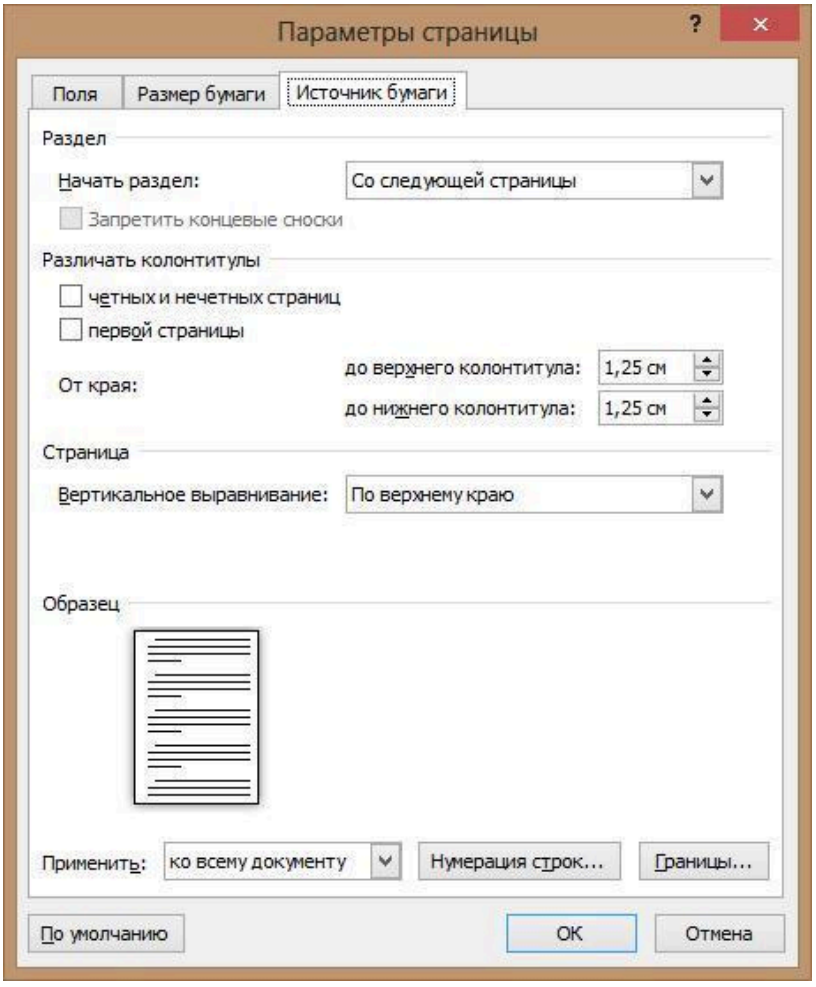
1. 

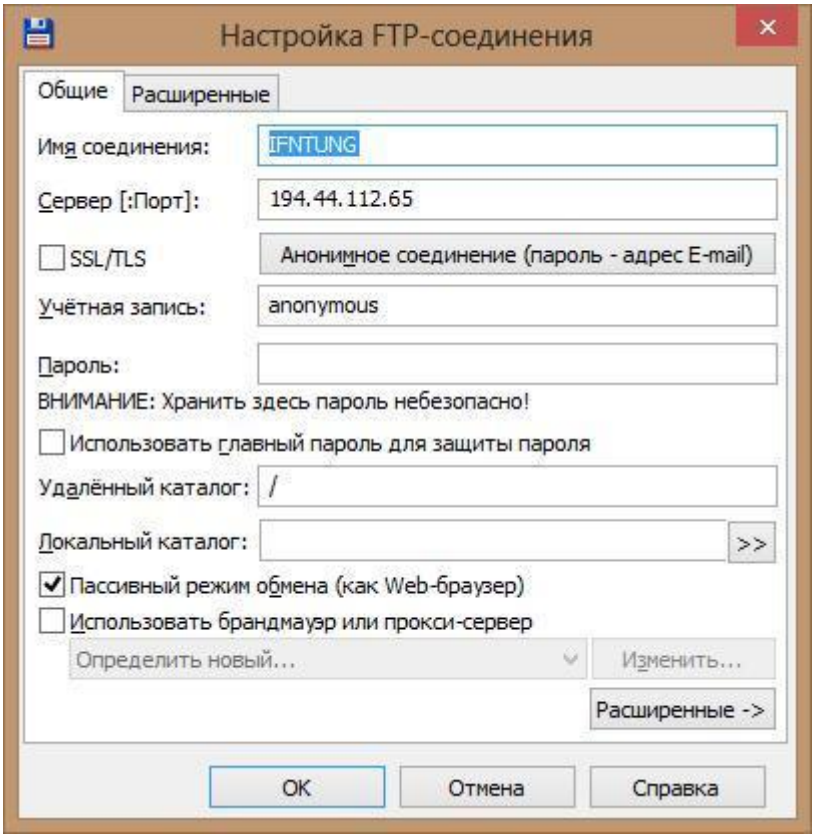
2. 

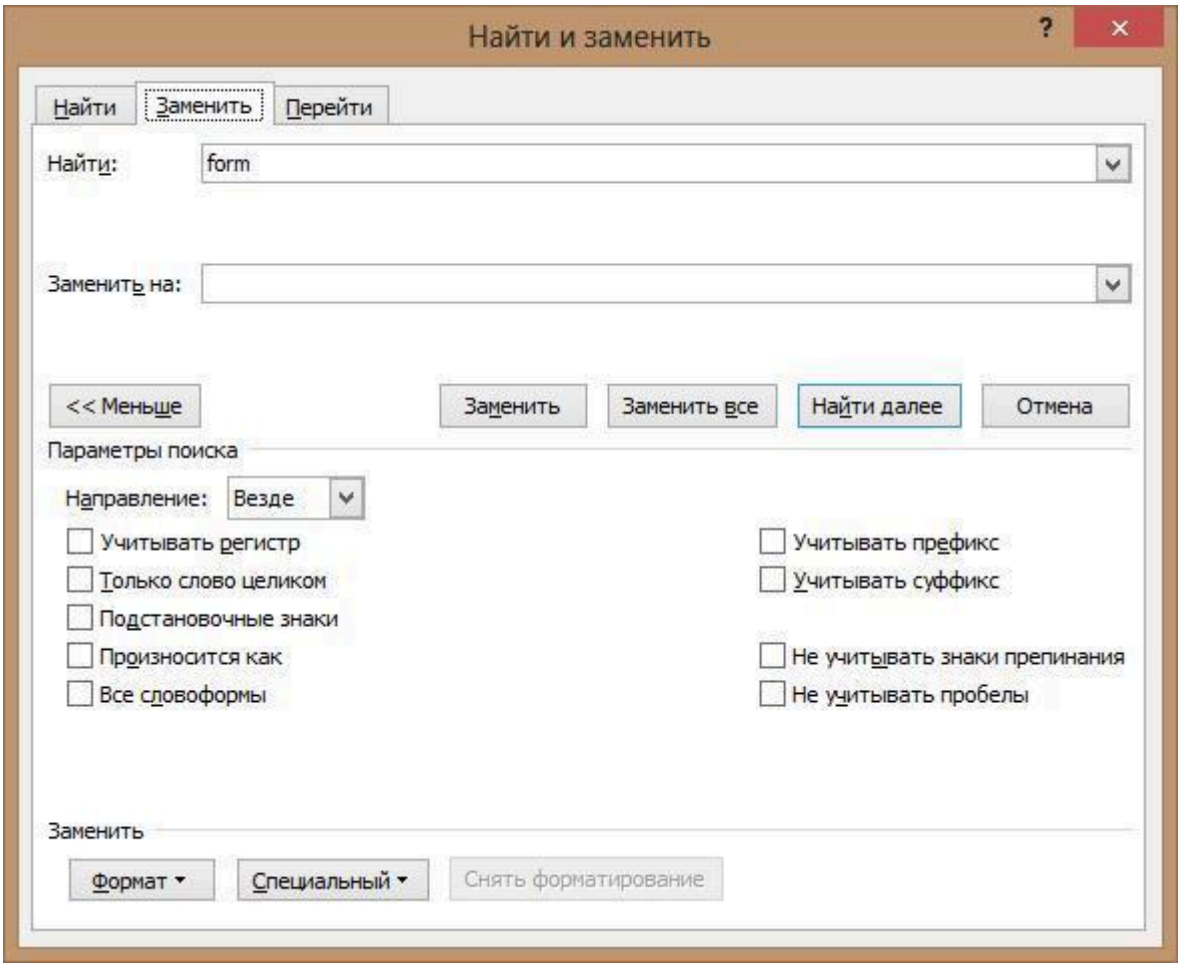
3. 
Абзац
Отступы и интервалы
Положение на странице
Общие
Выравнивание:
По левому краю
Уровень:
Основной текст
Отступ
Слева:
0 см
Справа:
0 см
первая строка:
(нет)
на:
Зеркальные отступы
Интервал
Перед:
0 пт
После:
10 пт
междустрочный:
Множитель
значение:
1,15
Не добавлять интервал между абзацами одного стиля
Образец
Табуляция...
По умолчанию
ОК
Отмена

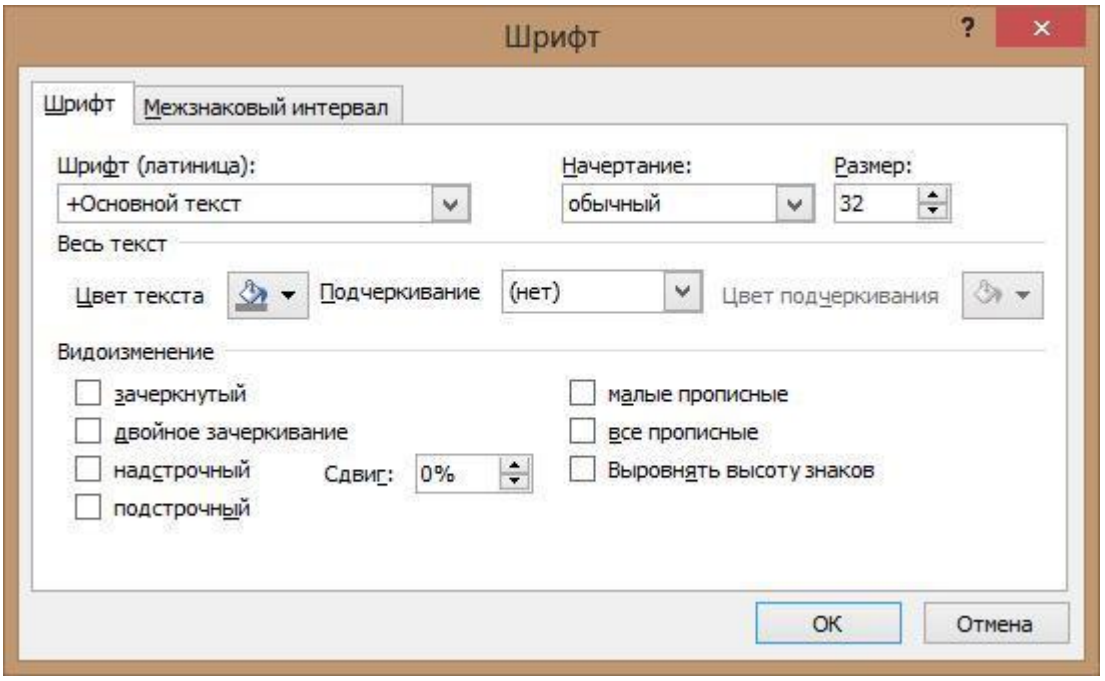
4. 
Параметры страницы
Поля
Размер бумаги
Источник бумаги
Поля
Верхнее:
1,5 см
Нижнее:
1,5 см
Левое:
2,5 см
Правое:
1,5 см
Переплет:
0 см
Положение переплета:
Слева
Ориентация
книжная
альбомная
Страницы
несколько страниц:
Обычный
Образец
Применить:
ко всему документу
По умолчанию
ОК
Отмена

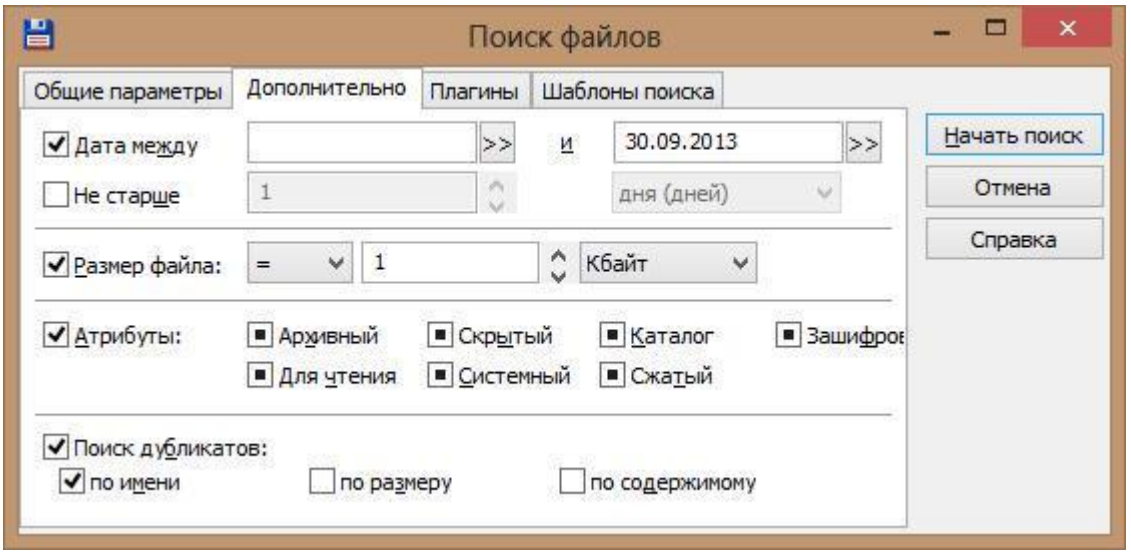
5. 
Параметры страницы
Поля
Размер бумаги
Источник бумаги
Размер бумаги:
A4 210 x 297 mm
Ширина:
21 см
Высота:
29,7 см
Подача бумаги
Первая страница:
По умолч. (Аркуш)
Аркуш
Аркуш (без полів)
Остальные страницы:
По умолч. (Аркуш)
Аркуш
Аркуш (без полів)
Образец
Применить:
ко всему документу
Параметры печати...
По умолчанию
OK
Отмена

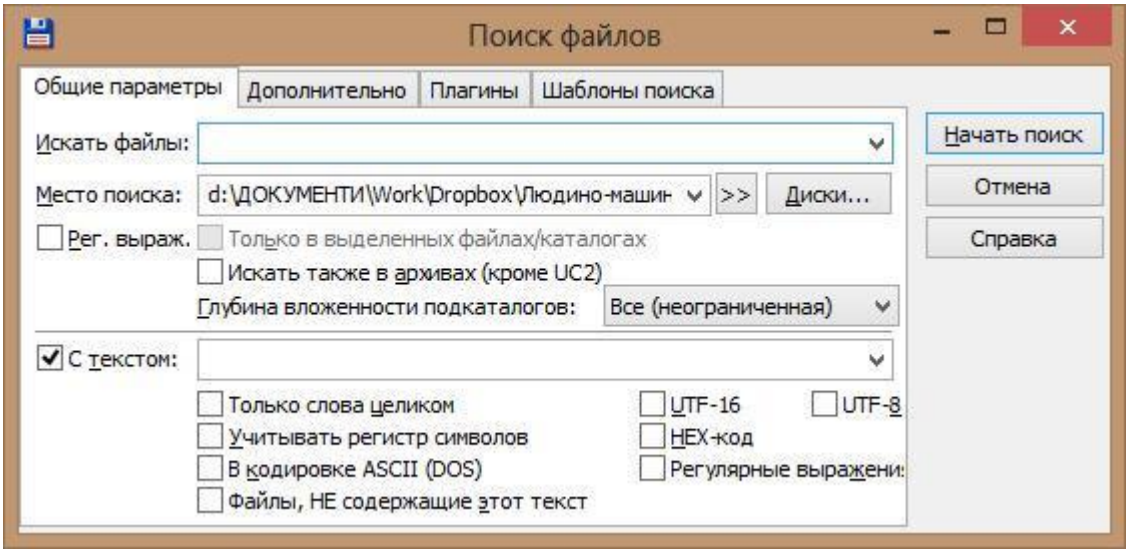
6. 
Параметры страницы
Поля
Размер бумаги
Источник бумаги
Раздел
Начать раздел:
Со следующей страницы
Запретить концевые сноски
Различать колонтитулы
четных и нечетных страниц
первой страницы
От края:
до верхнего колонтитула:
1,25 см
до нижнего колонтитула:
1,25 см
Страница
Вертикальное выравнивание:
По верхнему краю
Образец
Применить:
ко всему документу
Нумерация строк...
Границы...
По умолчанию
OK
Отмена

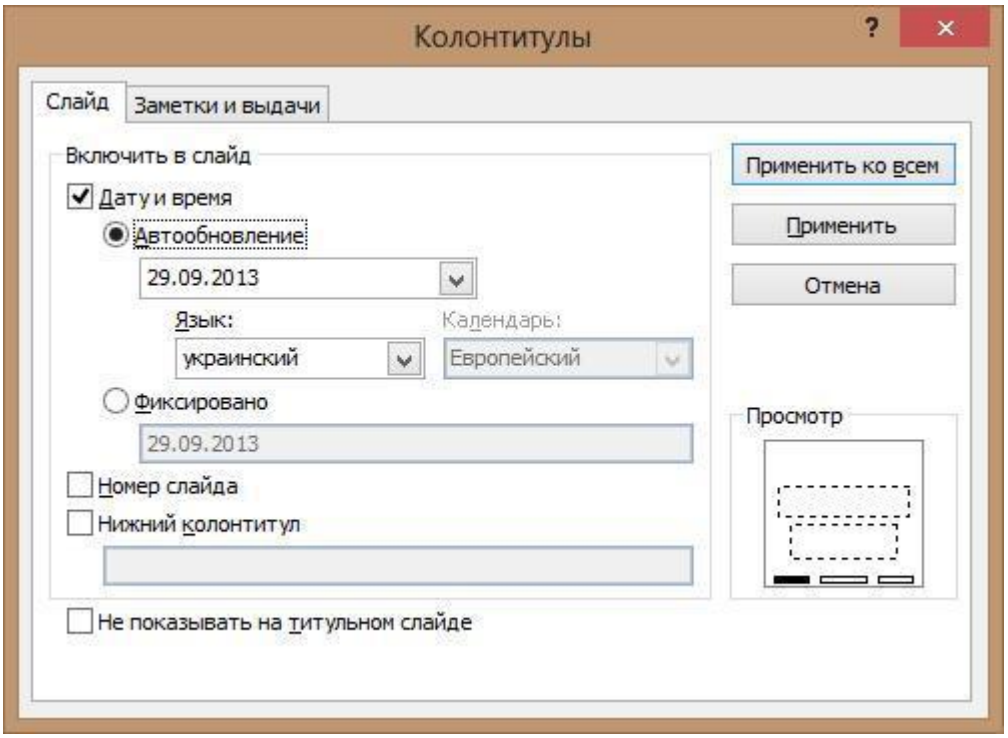
7. 
Настройка FTP-соединения
Общие
Расширенные
Имя соединения:
IFNTUNG
Сервер [:Порт]:
194.44.112.65
SSL/TLS
Анонимное соединение (пароль - адрес E-mail)
Учётная запись:
anonymous
Пароль:
ВНИМАНИЕ: Хранить здесь пароль небезопасно!
Использовать главный пароль для защиты пароля
Удалённый каталог:
/
Локальный каталог:
>>
Пассивный режим обмена (как Web-браузер)
Использовать брандмауэр или прокси-сервер
Определить новый...
Изменить...
Расширенные ->
OK
Отмена
Справка

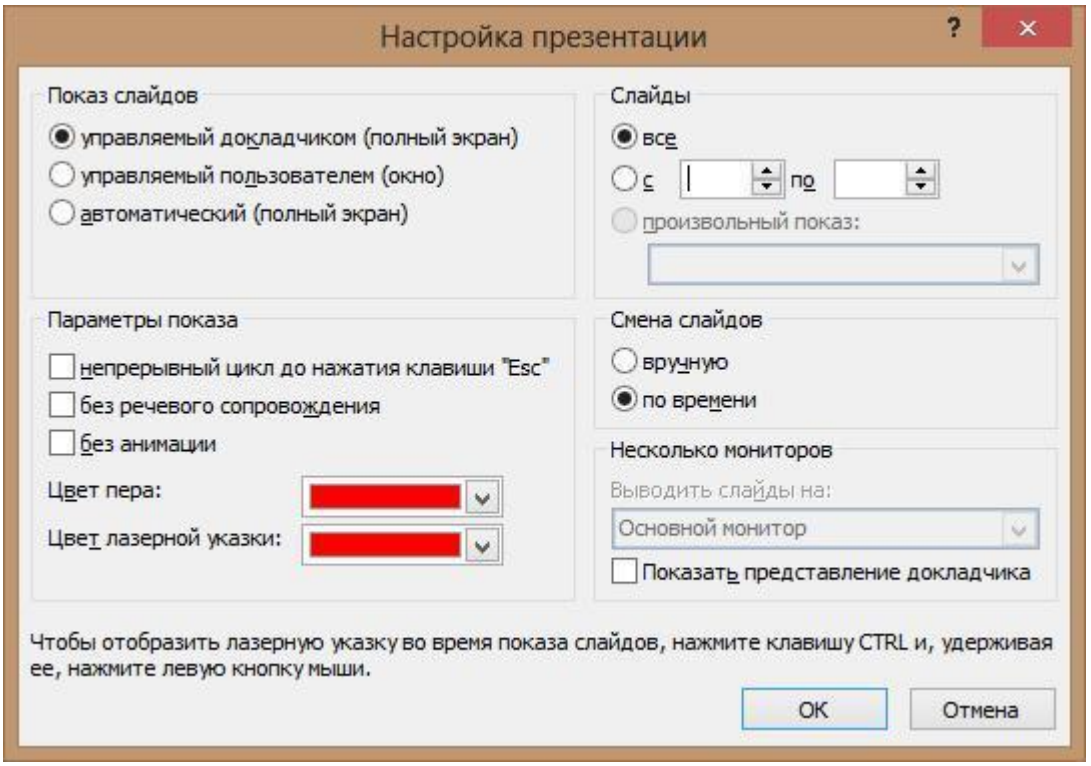
8. 
Найти и заменить
Найти
Заменить
Перейти
Найти:
form
Заменить на:
<< Меньше
Заменить
Заменить все
Найти далее
Отмена
Параметры поиска
Направление:
Везде
Учитывать регистр
Только слово целиком
Подстановочные знаки
Произносится как
Все словоформы
Учитывать префикс
Учитывать суффикс
Не учитывать знаки препинания
Не учитывать пробелы
Заменить
Формат
Специальный
Снять форматирование

9. 
Шрифт
Шрифт
Межзнаковый интервал
Шрифт (латиница):
+Основной текст
Начертание:
обычный
Размер:
32
Весь текст
Цвет текста
Подчеркивание
(нет)
Цвет подчеркивания
Видоизменение
зачеркнутый
двойное зачеркивание
надстрочный
подстрочный
Сдвиг:
0%
малые прописные
все прописные
Выровнять высоту знаков
OK
Отмена

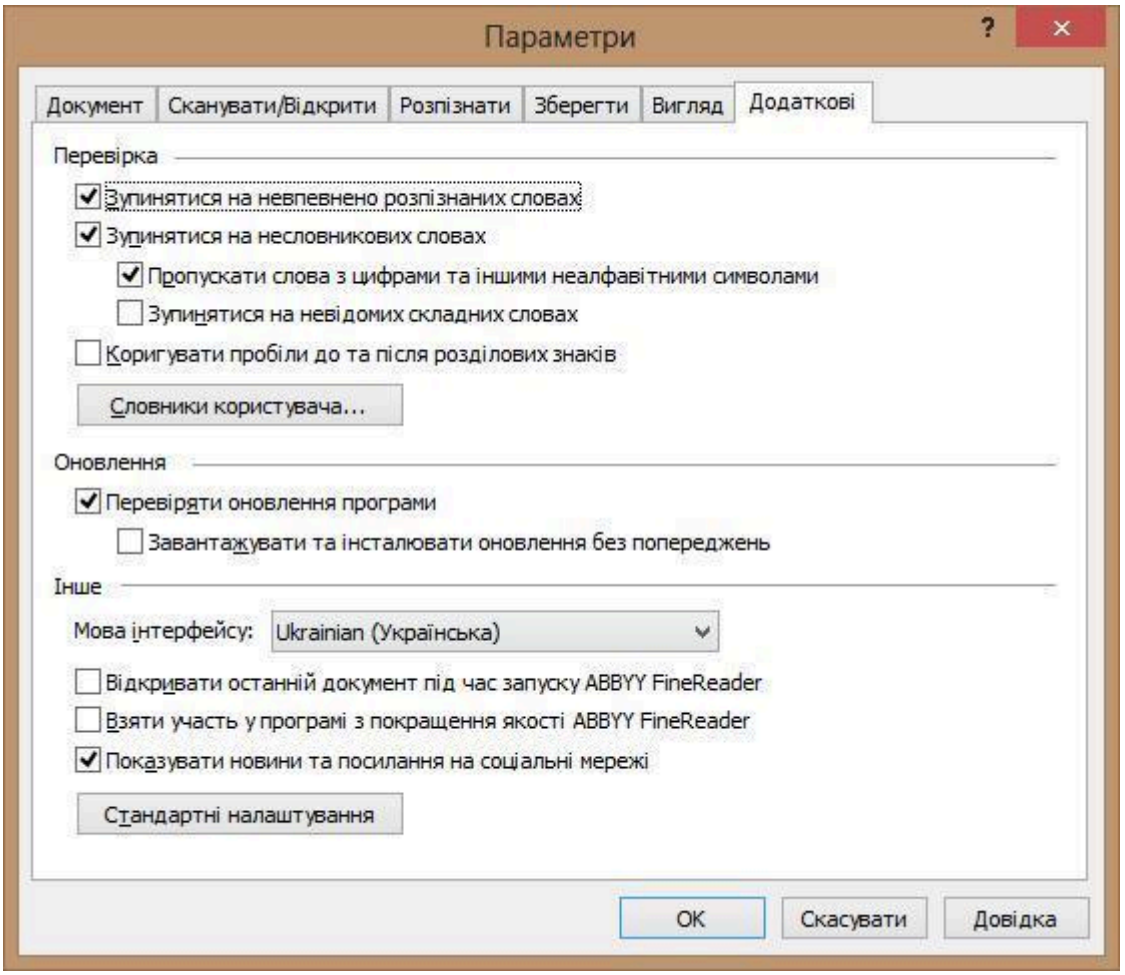
10. 
Поиск файлов
Общие параметры
Дополнительно
Плагины
Шаблоны поиска
Дата между
и
30.09.2013
Не старше
1
дня (дней)
Размер файла:
=
1
Кбайт
Атрибуты:
Архивный
Скрытый
Каталог
Для чтения
Системный
Сжатый
Поиск дубликатов:
по имени
по размеру
по содержимому
Начать поиск
Отмена
Справка

11. 
Поиск файлов
Общие параметры
Дополнительно
Плагины
Шаблоны поиска
Искать файлы:
Место поиска:
d:\ДОКУМЕНТИ\Work\Dropbox\Людино-машин
Диски...
Рег. выраж.
Только в выделенных файлах/каталогах
Искать также в архивах (кроме UC2)
Глубина вложенности подкаталогов:
Все (неограниченная)
С текстом:
Только слова целиком
Учитывать регистр символов
В кодировке ASCII (DOS)
Файлы, НЕ содержащие этот текст
UTF-16
UTF-8
HEX-код
Регулярные выражени
Начать поиск
Отмена
Справка

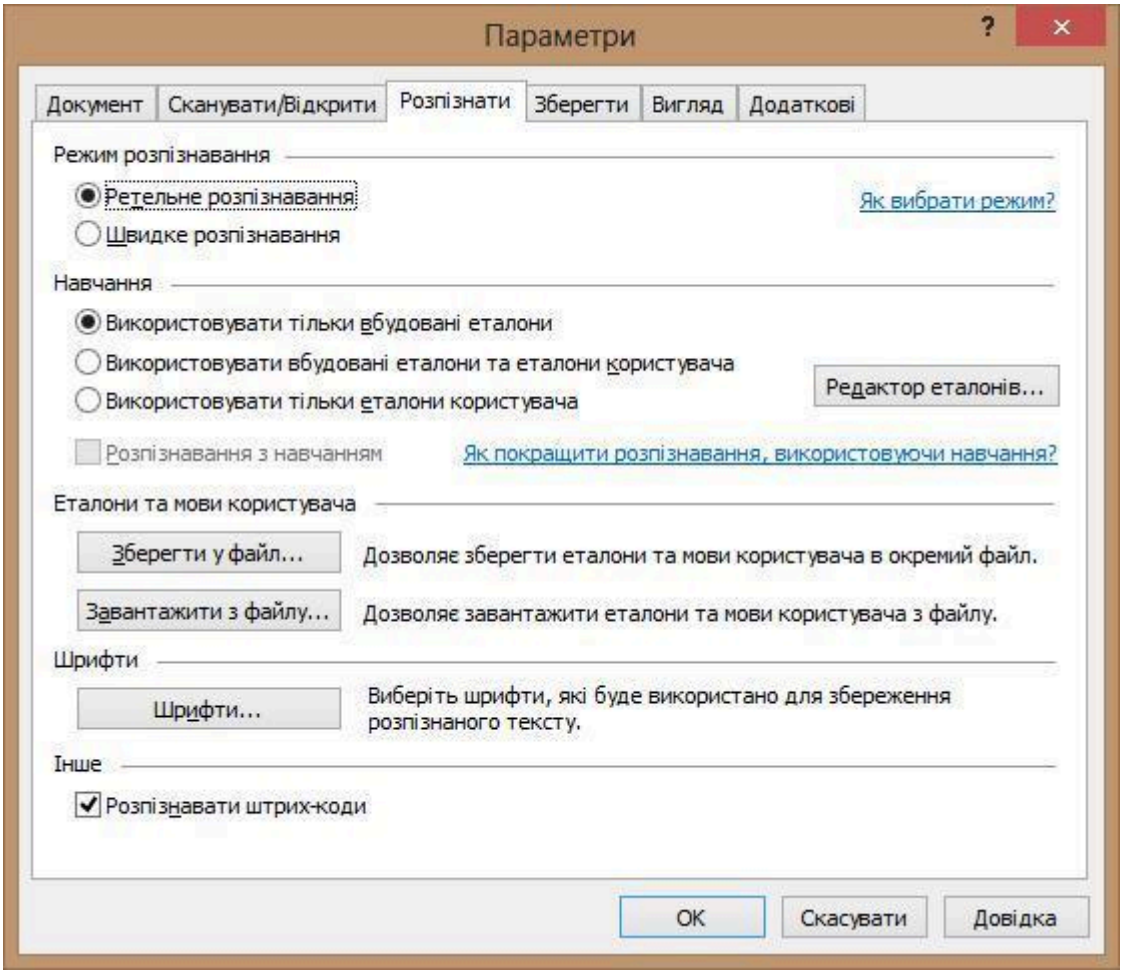
12. 
Колонтитулы
Слайд
Заметки и выдачи
Включить в слайд
Дату и время
Автообновление
29.09.2013
Язык:
украинский
Календарь:
Европейский
Фиксировано
29.09.2013
Номер слайда
Нижний колонтитул
Не показывать на титульном слайде
Применить ко всем
Применить
Отмена
Просмотр

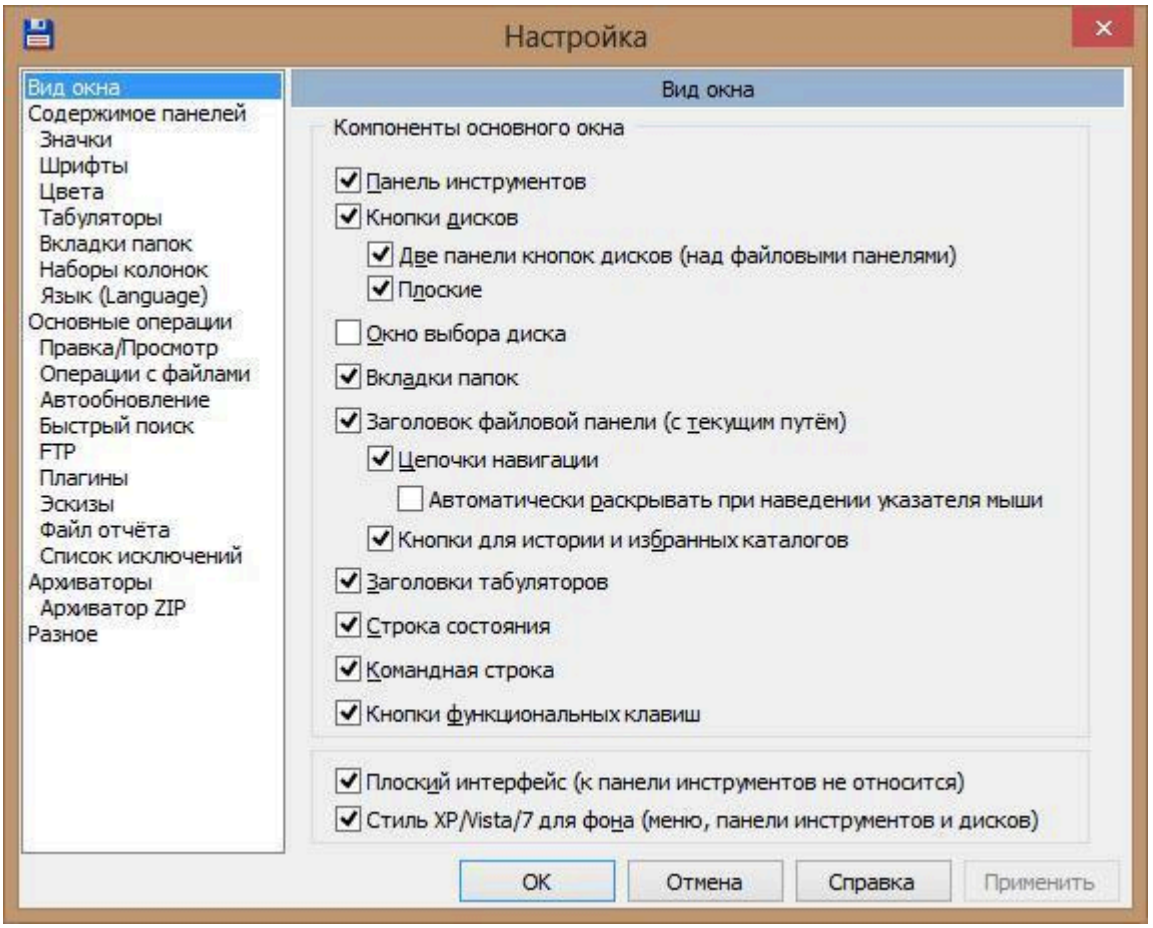
13. 
Настройка презентации
Показ слайдов
управляемый докладчиком (полный экран)
управляемый пользователем (окно)
автоматический (полный экран)
Слайды
все
с
по
произвольный показ:
Параметры показа
непрерывный цикл до нажатия клавиши "Esc"
без речевого сопровождения
без анимации
Цвет пера:
Цвет лазерной указки:
Смена слайдов
вручную
по времени
Несколько мониторов
Выводить слайды на:
Основной монитор
Показать представление докладчика
Чтобы отобразить лазерную указку во время показа слайдов, нажмите клавишу CTRL и, удерживая ее, нажмите левую кнопку мыши.
ОК
Отмена

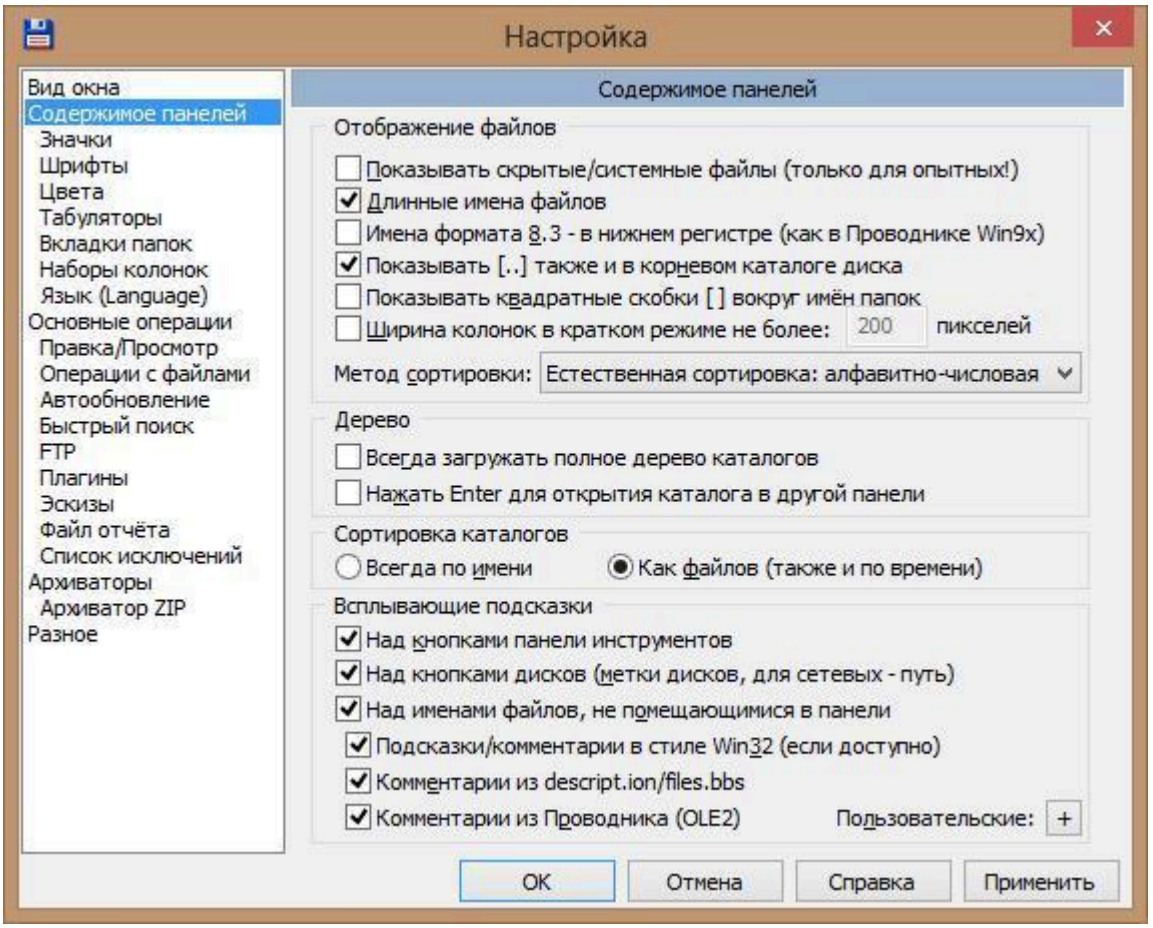
14.


Параметри
Документ
Сканувати/Відкрити
Розпізнати
Зберегти
Вигляд
Додаткові
Перевірка
Зупинятися на невпевнено розпізнаних словах
Зупинятися на несловникових словах
Пропускати слова з цифрами та іншими неалфавітними символами
Зупинятися на невідомих складних словах
Коригувати пробіли до та після розділових знаків
Словники користувача...
Оновлення
Перевіряти оновлення програми
Завантажувати та інсталювати оновлення без попереджень
Інше
Мова інтерфейсу:
Ukrainian (Українська)
Відкривати останній документ під час запуску ABBYY FineReader
Взяти участь у програмі з покращення якості ABBYY FineReader
Показувати новини та посилання на соціальні мережі
Стандартні налаштування
OK
Скасувати
Довідка

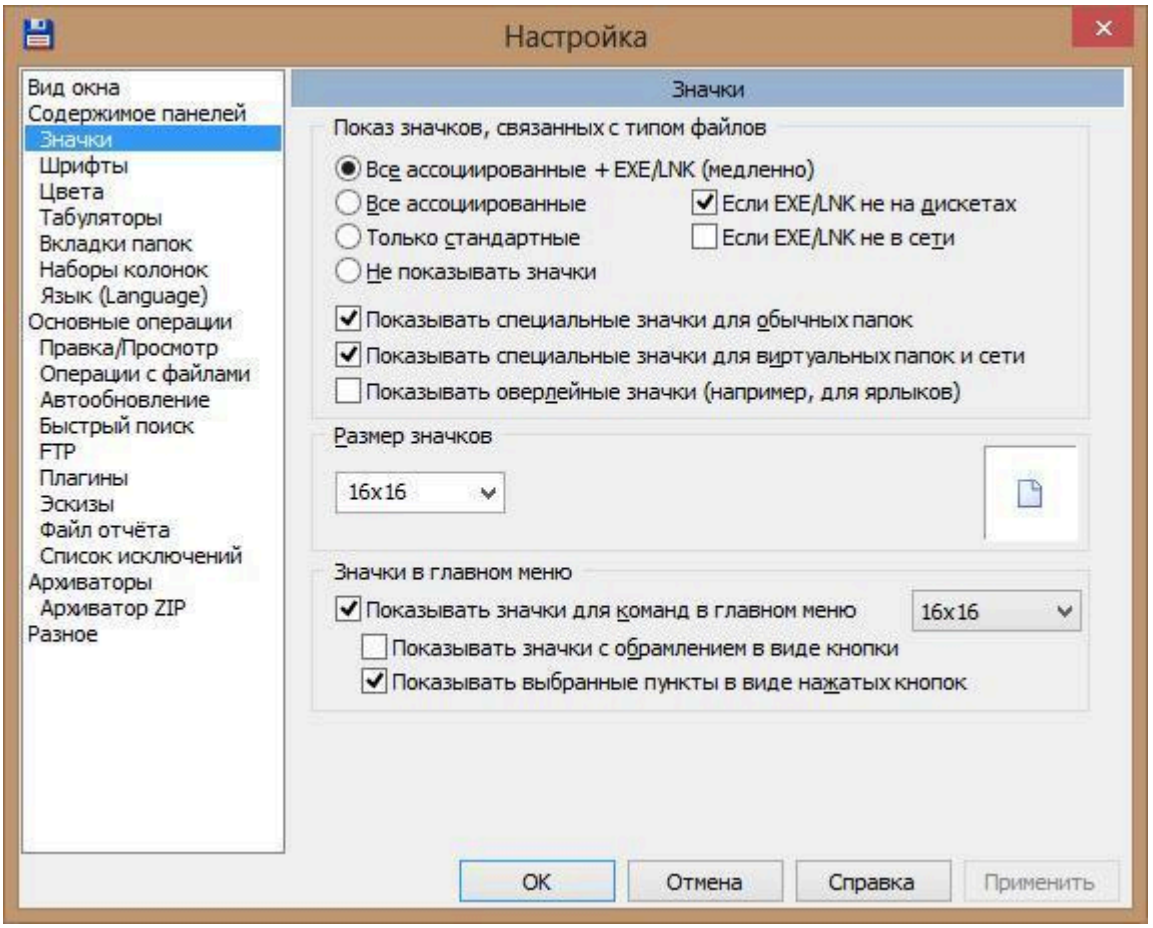
15.


Параметри
Документ
Сканувати/Відкрити
Розпізнати
Зберегти
Вигляд
Додаткові
Режим розпізнавання
Ретельне розпізнавання
Швидке розпізнавання
Як вибрати режим?
Навчання
Використовувати тільки вбудовані еталони
Використовувати вбудовані еталони та еталони користувача
Використовувати тільки еталони користувача
Редактор еталонів...
Розпізнавання з навчанням
Як покращити розпізнавання, використовуючи навчання?
Еталони та мови користувача
Зберегти у файл...
Дозволяє зберегти еталони та мови користувача в окремий файл.
Завантажити з файлу...
Дозволяє завантажити еталони та мови користувача з файлу.
Шрифти
Шрифти...
Виберіть шрифти, які буде використано для збереження розпізнаного тексту.
Інше
Розпізнавати штрих-коди
OK
Скасувати
Довідка

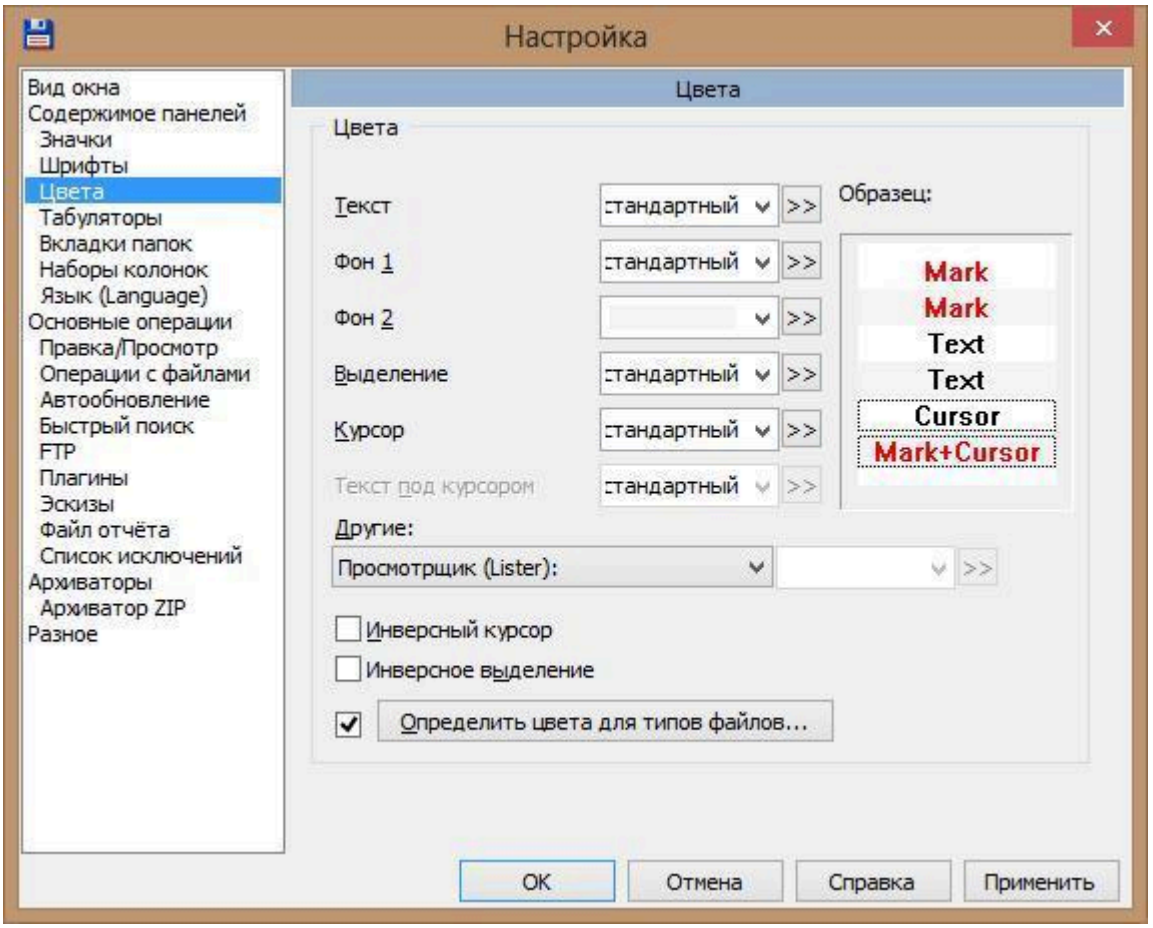
16. 
Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Вид окна
Компоненты основного окна
Панель инструментов
Кнопки дисков
Две панели кнопок дисков (над файловыми панелями)
Плоские
Окно выбора диска
Вкладки папок
Заголовок файловой панели (с текущим путём)
Цепочки навигации
Автоматически раскрывать при наведении указателя мыши
Кнопки для истории и избранных каталогов
Заголовки табуляторов
Строка состояния
Командная строка
Кнопки функциональных клавиш
Плоский интерфейс (к панели инструментов не относится)
Стиль XP/Vista/7 для фона (меню, панели инструментов и дисков)
OK
Отмена
Справка
Применить

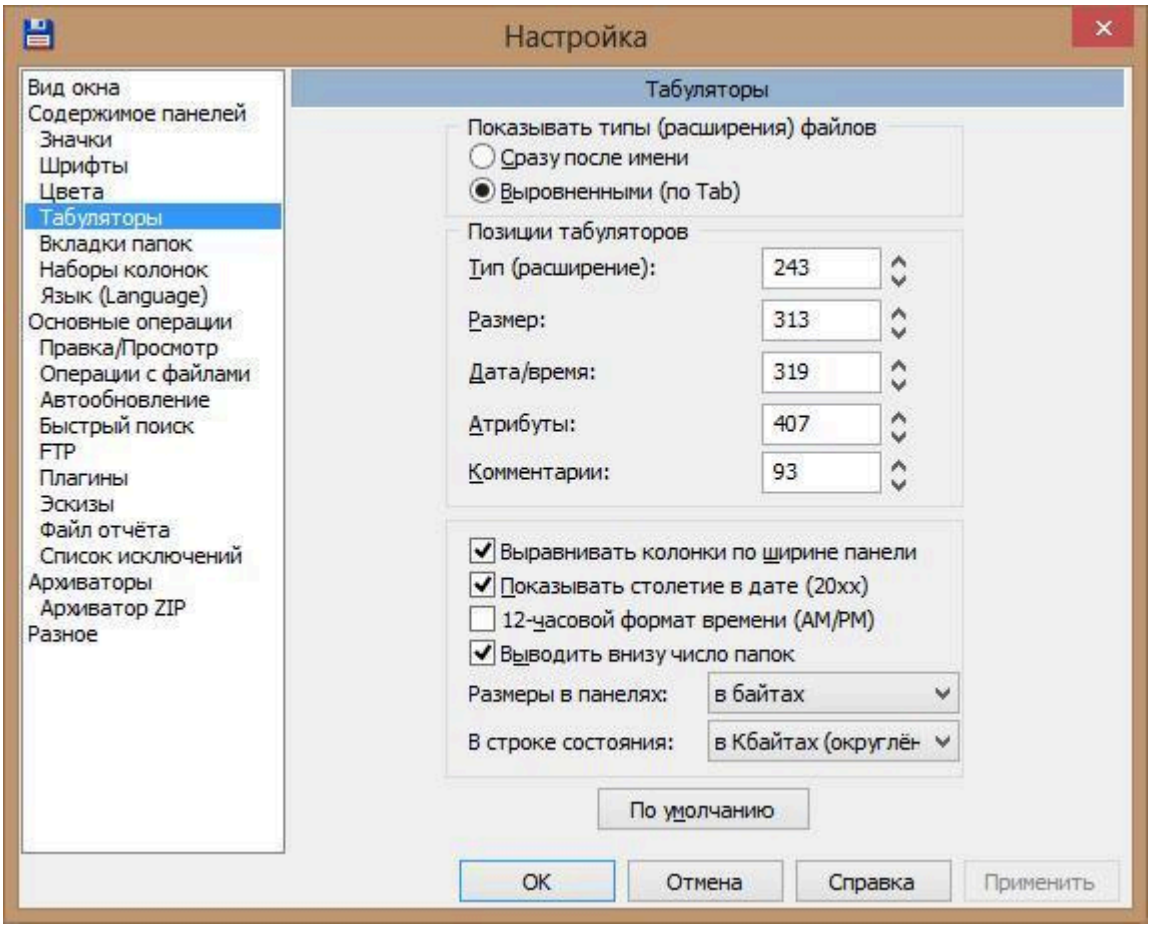
17. 
Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Содержимое панелей
Отображение файлов
Показывать скрытые/системные файлы (только для опытных!)
Длинные имена файлов
Имена формата 8.3 - в нижнем регистре (как в Проводнике Win9x)
Показывать [..] также и в корневом каталоге диска
Показывать квадратные скобки [ ] вокруг имён папок
Ширина колонок в кратком режиме не более:
200
пикселей
Метод сортировки:
Естественная сортировка: алфавитно-числовая
Дерево
Всегда загружать полное дерево каталогов
Нажать Enter для открытия каталога в другой панели
Сортировка каталогов
Всегда по имени
Как файлов (также и по времени)
Всплывающие подсказки
Над кнопками панели инструментов
Над кнопками дисков (метки дисков, для сетевых - путь)
Над именами файлов, не помещающимися в панели
Подсказки/комментарии в стиле Win32 (если доступно)
Комментарии из descript.ion/files.bbs
Комментарии из Проводника (OLE2)
Пользовательские:
+
OK
Отмена
Справка
Применить

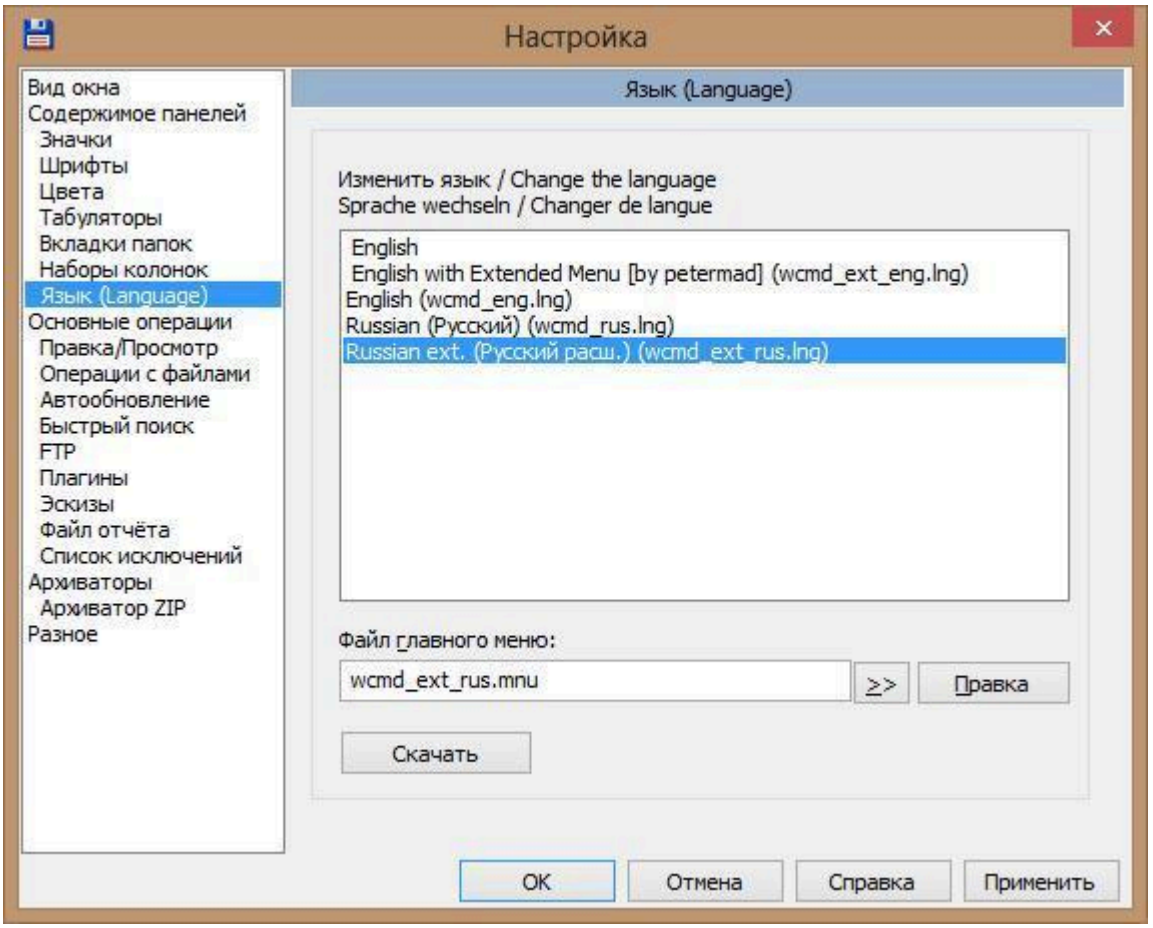
18.


Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Значки
Показ значков, связанных с типом файлов
Все ассоциированные + EXE/LNK (медленно)
Все ассоциированные
Только стандартные
Не показывать значки
Если EXE/LNK не на дискетах
Если EXE/LNK не в сети
Показывать специальные значки для обычных папок
Показывать специальные значки для виртуальных папок и сети
Показывать оверлейные значки (например, для ярлыков)
Размер значков
16x16
Значки в главном меню
Показывать значки для команд в главном меню
16x16
Показывать значки с обрамлением в виде кнопки
Показывать выбранные пункты в виде нажатых кнопок
OK
Отмена
Справка
Применить

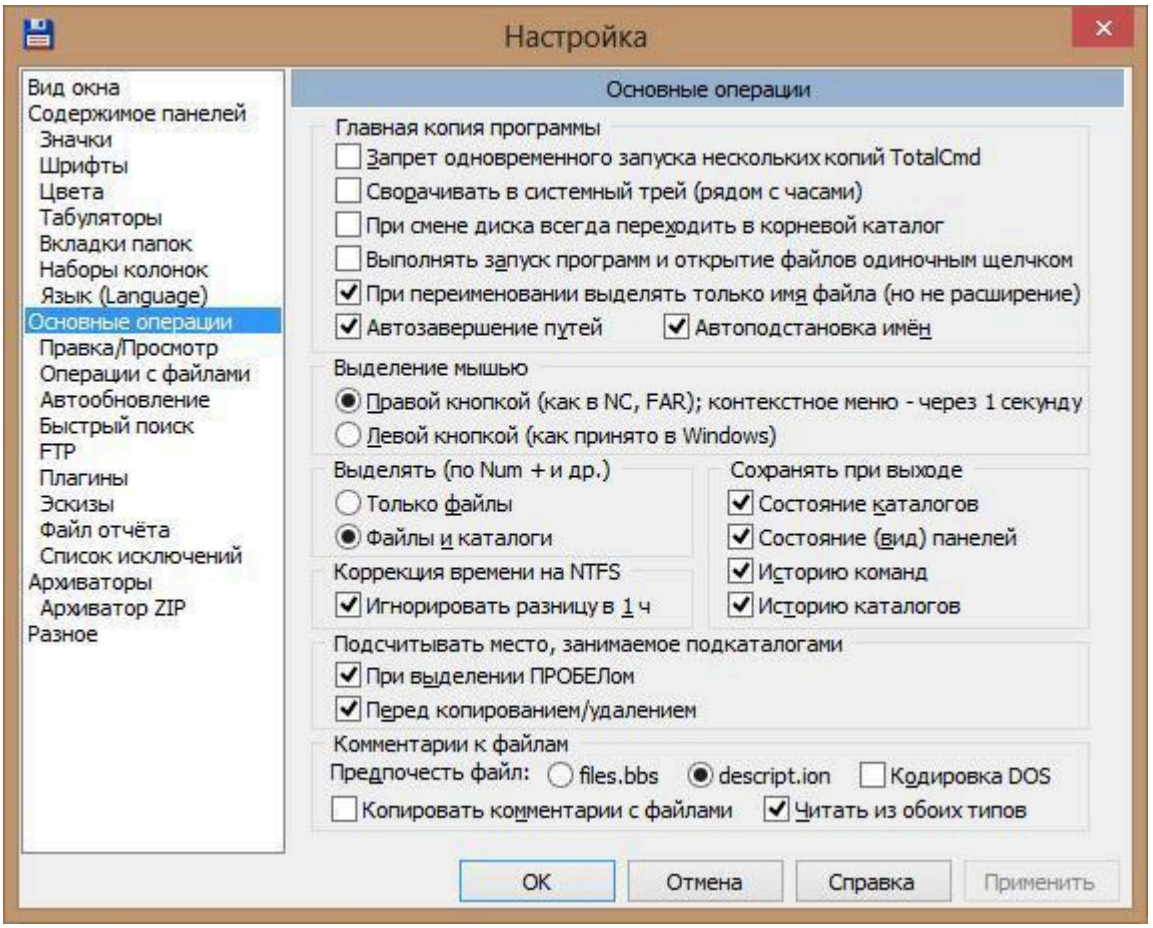
19.


Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Цвета
Цвета
Текст
Фон 1
Фон 2
Выделение
Курсор
Текст под курсором
стандартный
>>
Образец:
Mark
Mark
Text
Text
Cursor
Mark+Cursor
Другие:
Просмотрщик (Lister):
Инверсный курсор
Инверсное выделение
Определить цвета для типов файлов...
OK
Отмена
Справка
Применить

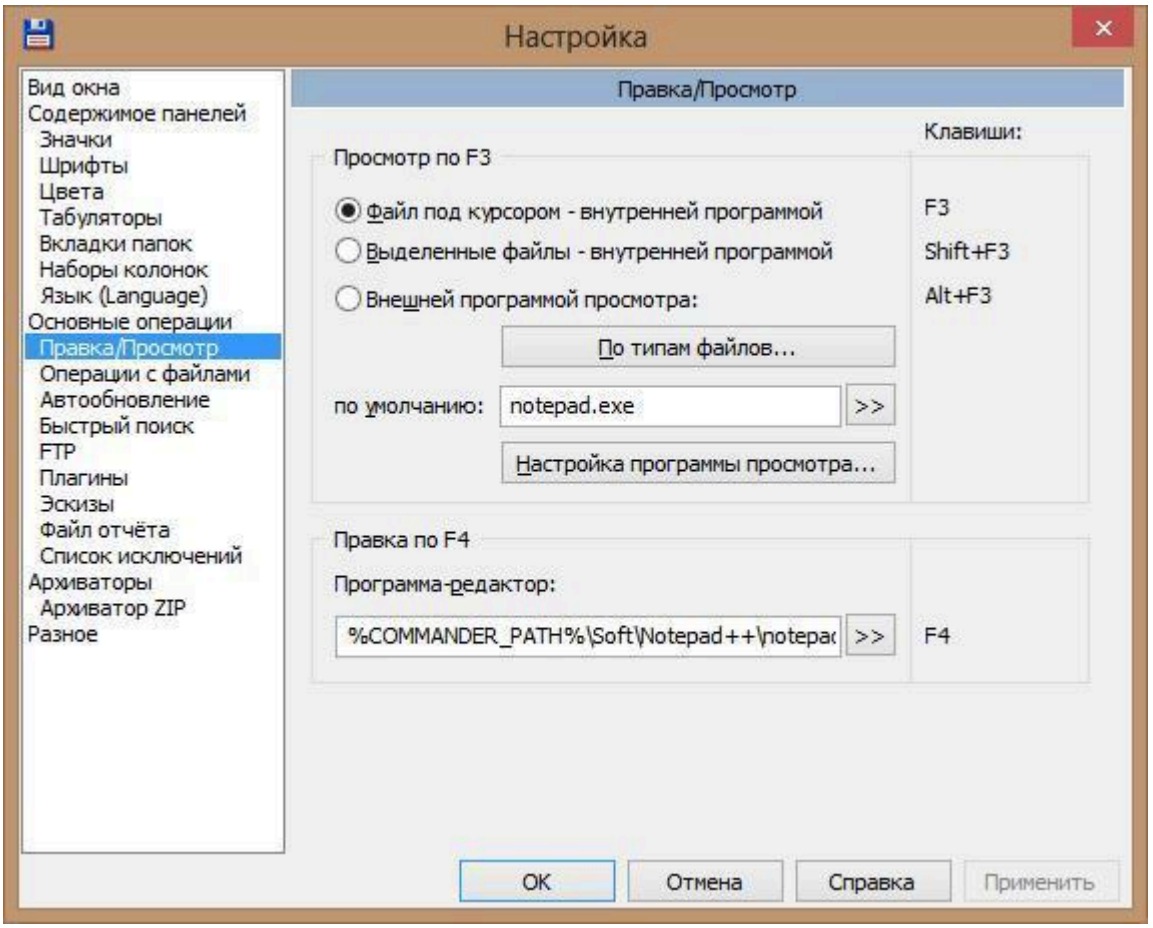
20. 
Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Табуляторы
Показывать типы (расширения) файлов
Сразу после имени
Выровненными (по Tab)
Позиции табуляторов
Тип (расширение): 243
Размер: 313
Дата/время: 319
Атрибуты: 407
Комментарии: 93
Выравнивать колонки по ширине панели
Показывать столетие в дате (20xx)
12-часовой формат времени (AM/PM)
Выводить внизу число папок
Размеры в панелях: в байтах
В строке состояния: в Кбайтах (округлён
По умолчанию
OK
Отмена
Справка
Применить

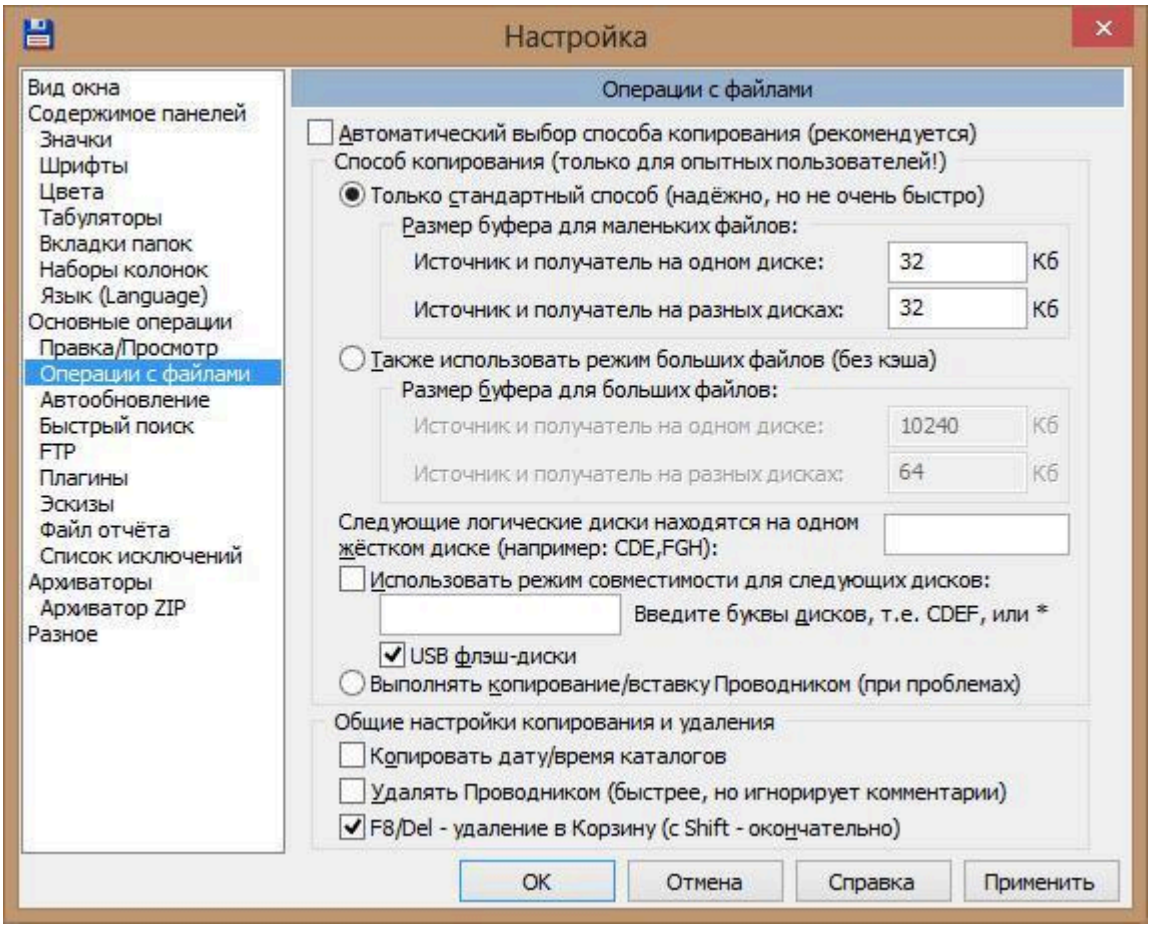
21. 
Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Язык (Language)
Изменить язык / Change the language
Sprache wechseln / Changer de langue
English
English with Extended Menu [by petermad] (wcmd_ext_eng.lng)
English (wcmd_eng.lng)
Russian (Русский) (wcmd_rus.lng)
Russian ext. (Русский расш.) (wcmd_ext_rus.lng)
Файл главного меню:
wcmd_ext_rus.mnu
>>
Правка
Скачать
OK
Отмена
Справка
Применить

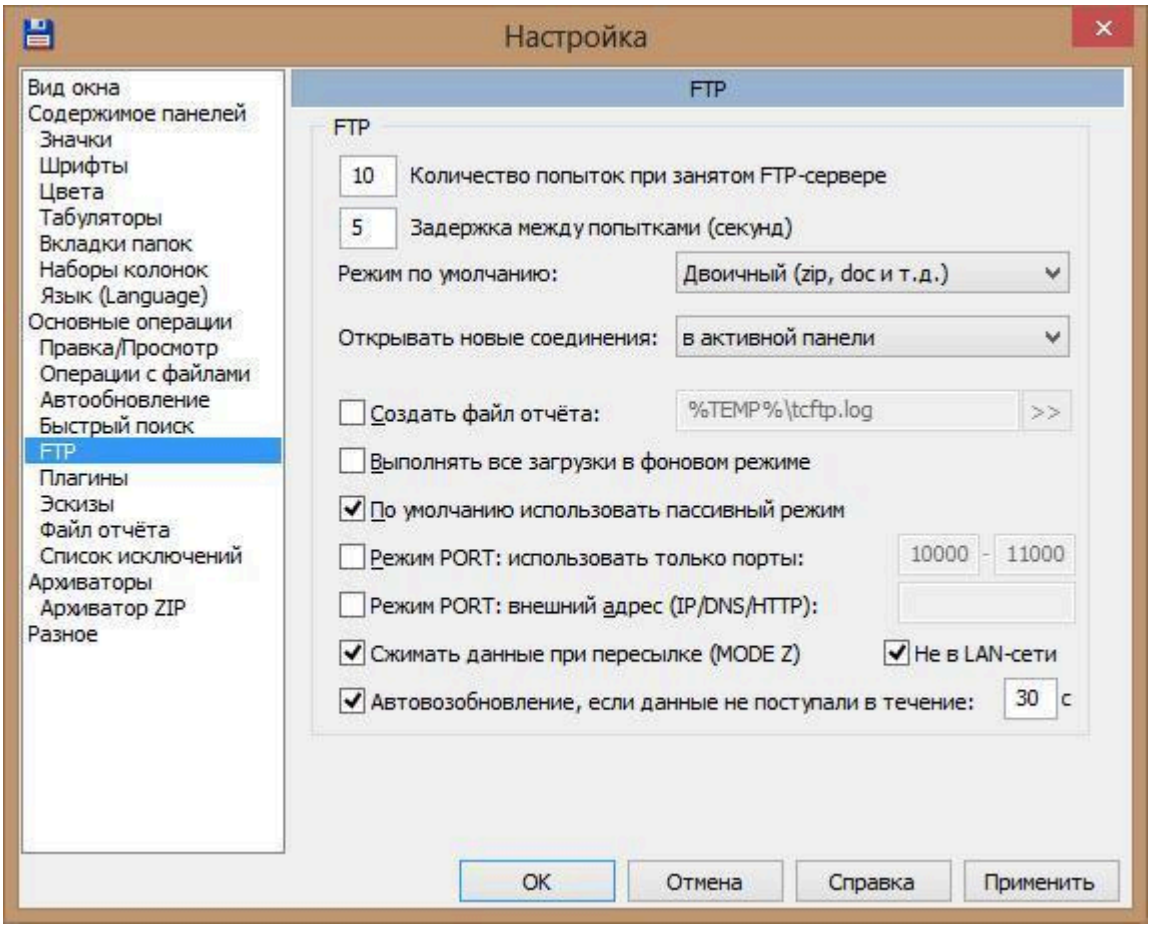
22.


Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Основные операции
Главная копия программы
Запрет одновременного запуска нескольких копий TotalCmd
Сворачивать в системный трей (рядом с часами)
При смене диска всегда переходить в корневой каталог
Выполнять запуск программ и открытие файлов одиночным щелчком
При переименовании выделять только имя файла (но не расширение)
Автозавершение путей
Автоподстановка имён
Выделение мышью
Правой кнопкой (как в NC, FAR); контекстное меню - через 1 секунду
Левой кнопкой (как принято в Windows)
Выделять (по Num + и др.)
Только файлы
Файлы и каталоги
Сохранять при выходе
Состояние каталогов
Состояние (вид) панелей
Историю команд
Историю каталогов
Коррекция времени на NTFS
Игнорировать разницу в 1 ч
Подсчитывать место, занимаемое подкаталогами
При выделении ПРОБЕЛом
Перед копированием/удалением
Комментарии к файлам
Предпочесть файл:
files.bbs
descript.ion
Кодировка DOS
Копировать комментарии с файлами
Читать из обоих типов
OK
Отмена
Справка
Применить

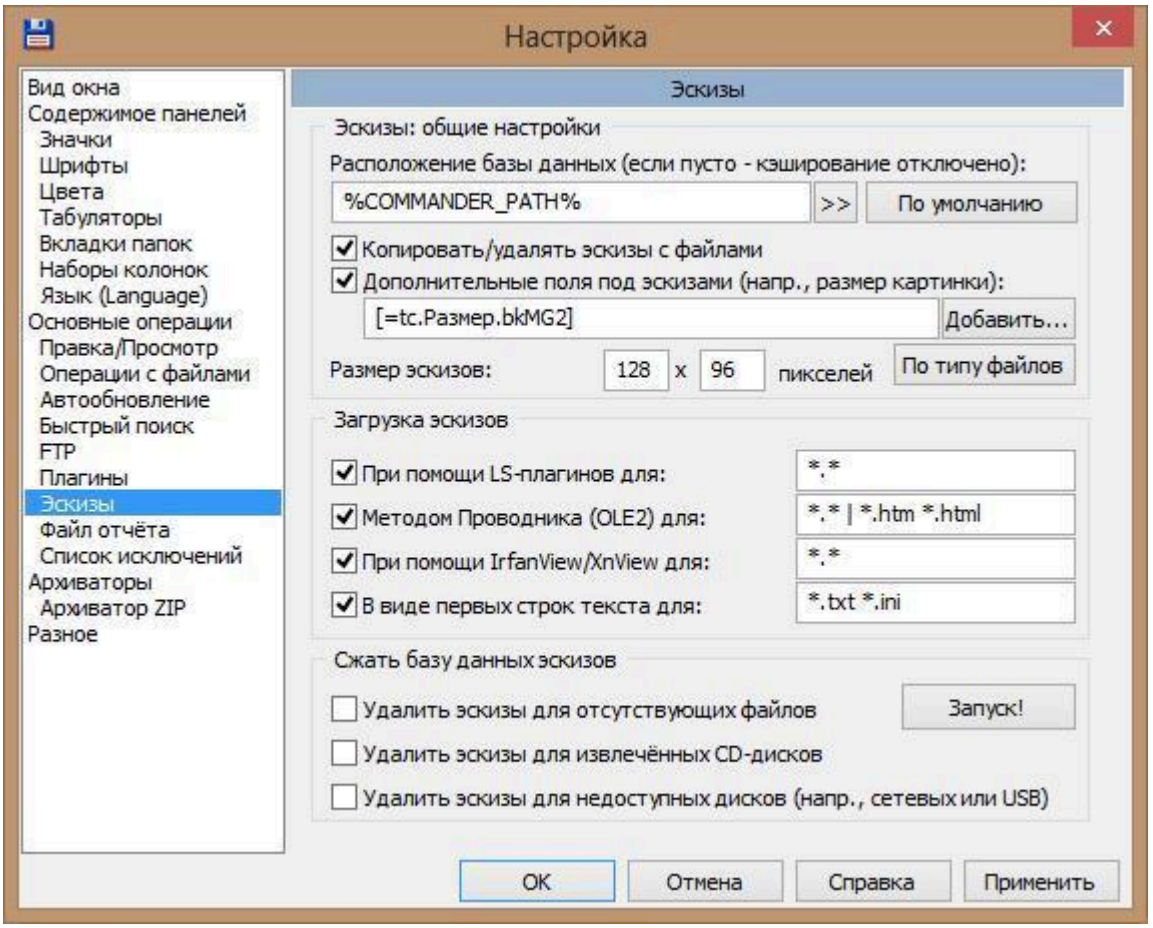
23.


Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Правка/Просмотр
Клавиши:
Просмотр по F3
Файл под курсором - внутренней программой
F3
Выделенные файлы - внутренней программой
Shift+F3
Внешней программой просмотра:
Alt+F3
По типам файлов...
по умолчанию:
notepad.exe
>>
Настройка программы просмотра...
Правка по F4
Программа-редактор:
%COMMANDER_PATH%\Soft\Notepad++\notepac
>>
F4
OK
Отмена
Справка
Применить

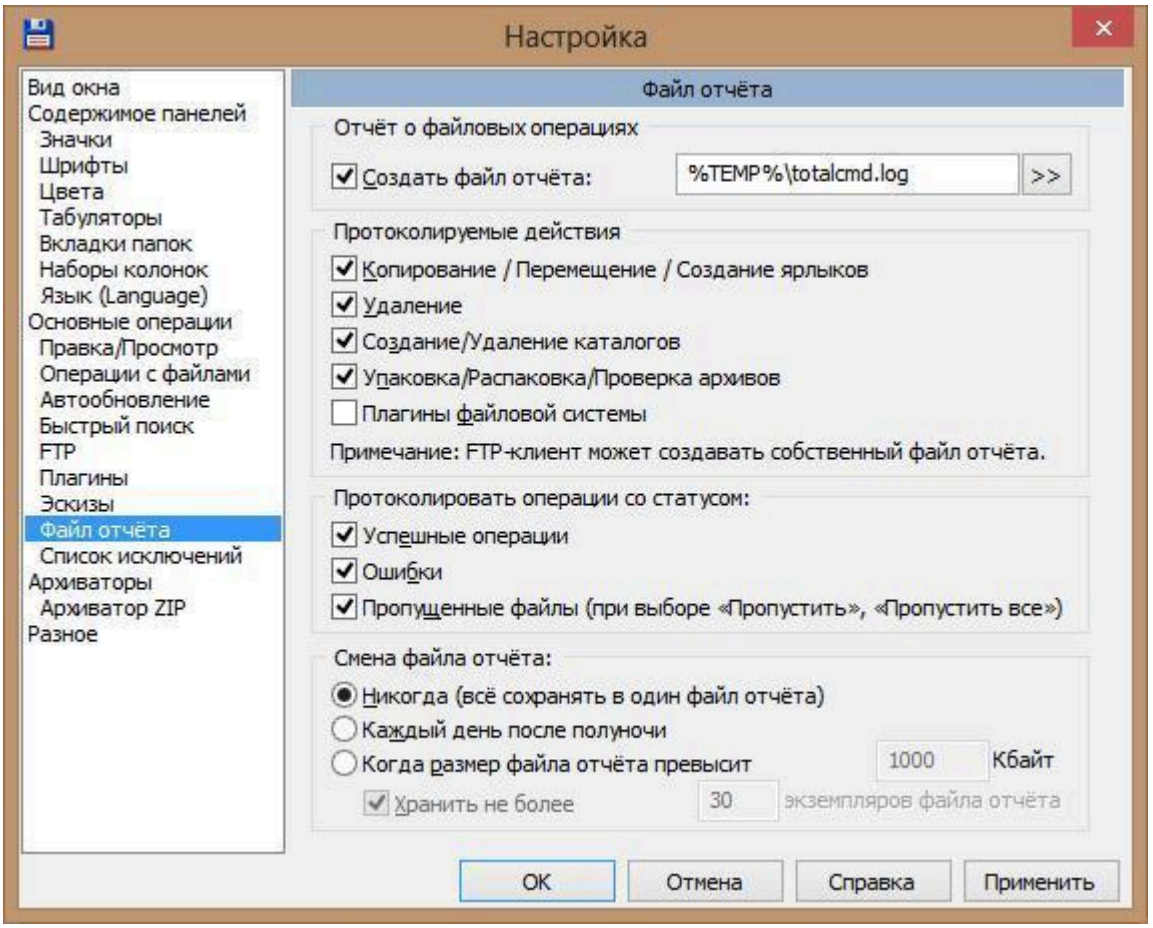
24. 
Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Операции с файлами
Автоматический выбор способа копирования (рекомендуется)
Способ копирования (только для опытных пользователей!)
Только стандартный способ (надёжно, но не очень быстро)
Размер буфера для маленьких файлов:
Источник и получатель на одном диске: 32 Кб
Источник и получатель на разных дисках: 32 Кб
Также использовать режим больших файлов (без кэша)
Размер буфера для больших файлов:
Источник и получатель на одном диске: 10240 Кб
Источник и получатель на разных дисках: 64 Кб
Следующие логические диски находятся на одном жёстком диске (например: CDE,FGH):
Использовать режим совместимости для следующих дисков:
Введите буквы дисков, т.е. CDEF, или *
USB флэш-диски
Выполнять копирование/вставку Проводником (при проблемах)
Общие настройки копирования и удаления
Копировать дату/время каталогов
Удалять Проводником (быстрее, но игнорирует комментарии)
F8/Del - удаление в Корзину (с Shift - окончательно)
OK
Отмена
Справка
Применить

25. 
Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
FTP
FTP
10 Количество попыток при занятом FTP-сервере
5 Задержка между попытками (секунд)
Режим по умолчанию: Двоичный (zip, doc и т.д.)
Открывать новые соединения: в активной панели
Создать файл отчёта: %TEMP%\tcftp.log >>
Выполнять все загрузки в фоновом режиме
По умолчанию использовать пассивный режим
Режим PORT: использовать только порты: 10000 - 11000
Режим PORT: внешний адрес (IP/DNS/HTTP):
Сжимать данные при пересылке (MODE Z)
Не в LAN-сети
Автовозобновление, если данные не поступали в течение: 30 с
OK
Отмена
Справка
Применить

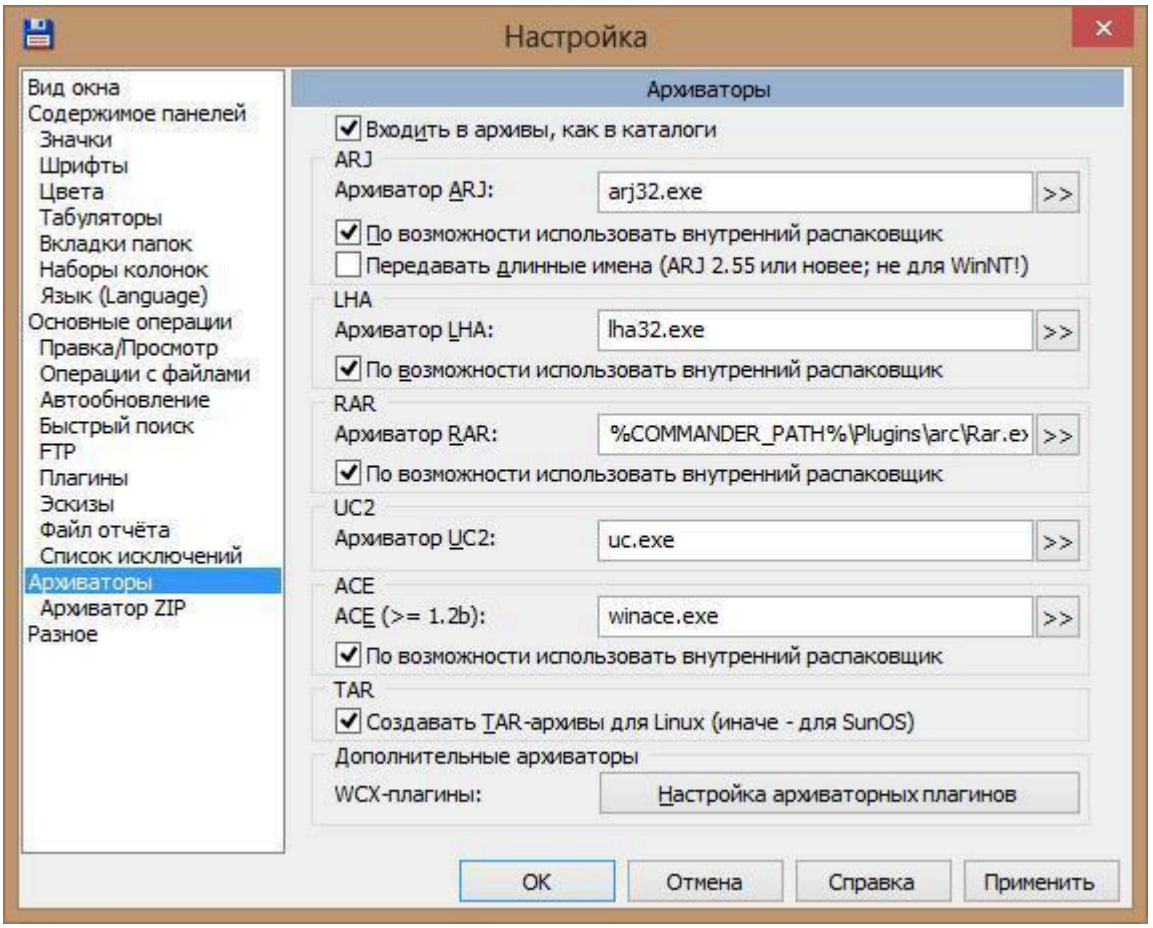
26.


Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Эскизы
Эскизы: общие настройки
Расположение базы данных (если пусто - кэширование отключено):
%COMMANDER_PATH%
>>
По умолчанию
Копировать/удалять эскизы с файлами
Дополнительные поля под эскизами (напр., размер картинки):
[=tc.Размер.bkMG2]
Добавить...
Размер эскизов:
128
x
96
пикселей
По типу файлов
Загрузка эскизов
При помощи LS-плагинов для:
.
Методом Проводника (OLE2) для:
. | *.htm *.html
При помощи IrfanView/XnView для:
.
В виде первых строк текста для:
*.txt *.ini
Сжать базу данных эскизов
Удалить эскизы для отсутствующих файлов
Запуск!
Удалить эскизы для извлечённых CD-дисков
Удалить эскизы для недоступных дисков (напр., сетевых или USB)
OK
Отмена
Справка
Применить

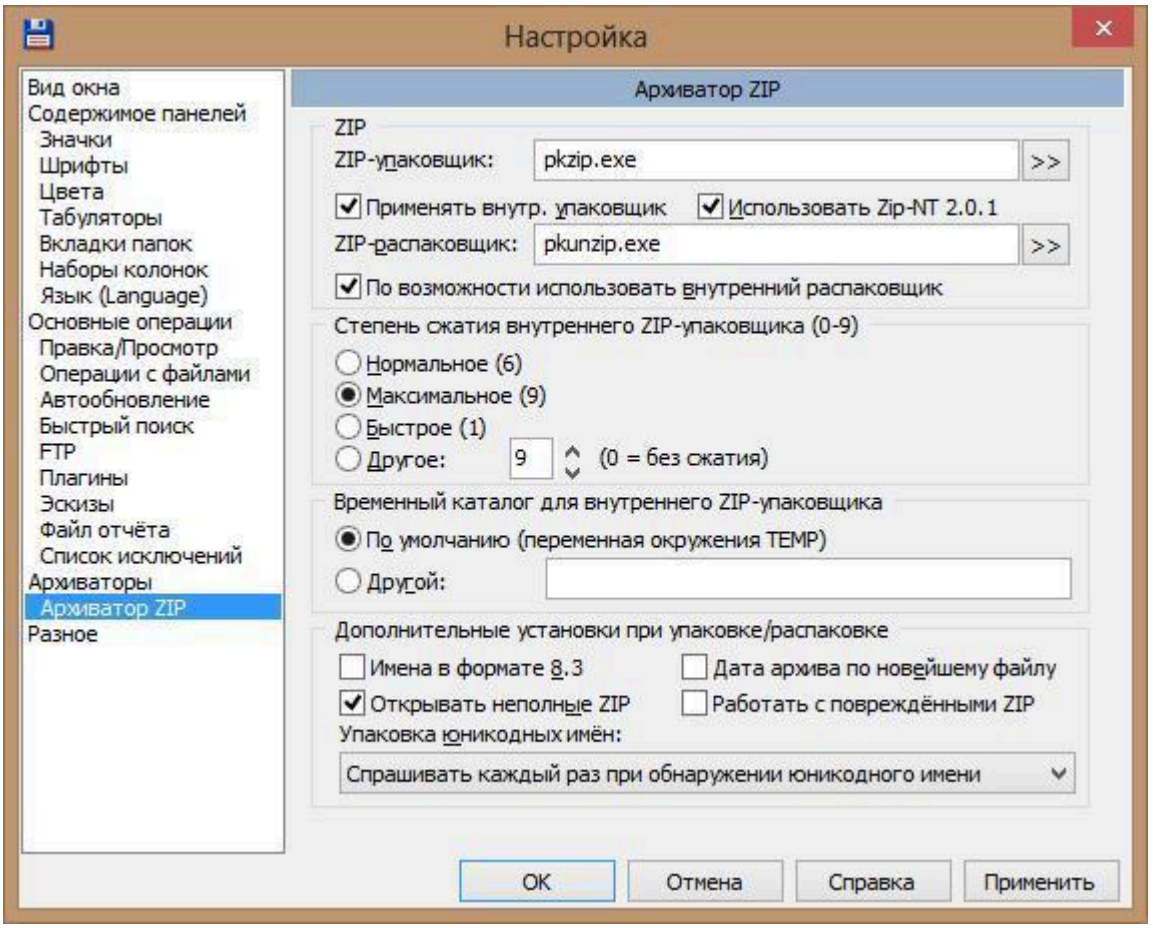
27.

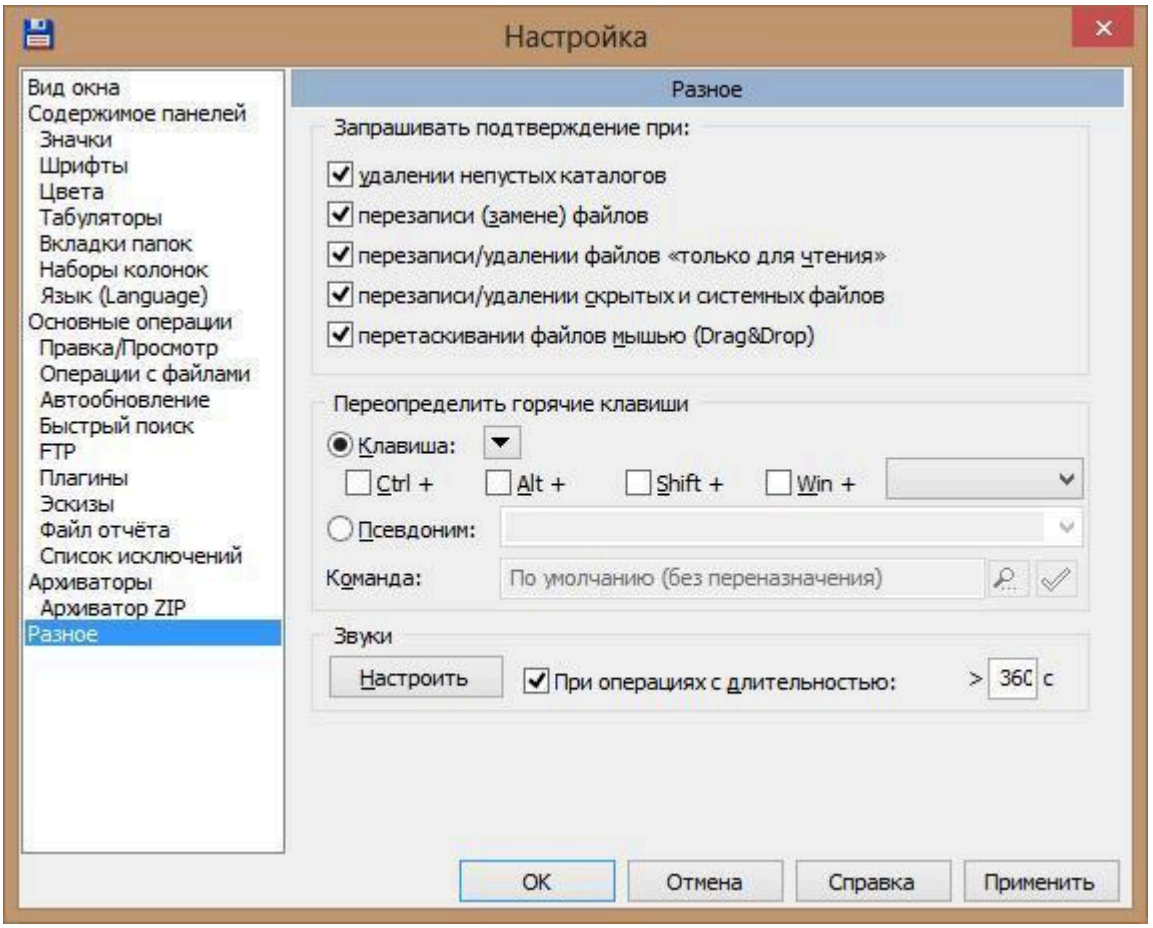

Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Файл отчёта
Отчёт о файловых операциях
Создать файл отчёта:
%TEMP%\totalcmd.log
>>
Протоколируемые действия
Копирование / Перемещение / Создание ярлыков
Удаление
Создание/Удаление каталогов
Упаковка/Распаковка/Проверка архивов
Плагины файловой системы
Примечание: FTP-клиент может создавать собственный файл отчёта.
Протоколировать операции со статусом:
Успешные операции
Ошибки
Пропущенные файлы (при выборе «Пропустить», «Пропустить все»)
Смена файла отчёта:
Никогда (всё сохранять в один файл отчёта)
Каждый день после полуночи
Когда размер файла отчёта превысит
1000
Кбайт
Хранить не более
30
экземпляров файла отчёта
OK
Отмена
Справка
Применить

28.


Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Архиваторы
Входить в архивы, как в каталоги
ARJ
Архиватор ARJ:
arj32.exe
>>
По возможности использовать внутренний распаковщик
Передавать длинные имена (ARJ 2.55 или новее; не для WinNT!)
LHA
Архиватор LHA:
lha32.exe
>>
По возможности использовать внутренний распаковщик
RAR
Архиватор RAR:
%COMMANDER_PATH%\Plugins\arc\Rar.e
>>
По возможности использовать внутренний распаковщик
UC2
Архиватор UC2:
uc.exe
>>
ACE
ACE (>= 1.2b):
winace.exe
>>
По возможности использовать внутренний распаковщик
TAR
Создавать TAR-архивы для Linux (иначе - для SunOS)
Дополнительные архиваторы
WCX-плагины:
Настройка архиваторных плагинов
OK
Отмена
Справка
Применить

29.


Настройка
Вид окна
Содержимое панелей
Значки
Шрифты
Цвета
Табуляторы
Вкладки папок
Наборы колонок
Язык (Language)
Основные операции
Правка/Просмотр
Операции с файлами
Автообновление
Быстрый поиск
FTP
Плагины
Эскизы
Файл отчёта
Список исключений
Архиваторы
Архиватор ZIP
Разное
Архиватор ZIP
ZIP
ZIP-упаковщик:
pkzip.exe
>>
Применять внутр. упаковщик
Использовать Zip-NT 2.0.1
ZIP-распаковщик:
pkunzip.exe
>>
По возможности использовать внутренний распаковщик
Степень сжатия внутреннего ZIP-упаковщика (0-9)
Нормальное (6)
Максимальное (9)
Быстрое (1)
Другое:
9
(0 = без сжатия)
Временный каталог для внутреннего ZIP-упаковщика
По умолчанию (переменная окружения TEMP)
Другой:
Дополнительные установки при упаковке/распаковке
Имена в формате 8.3
Дата архива по новейшему файлу
Открывать неполные ZIP
Работать с повреждёнными ZIP
Упаковка юникодных имён:
Спрашивать каждый раз при обнаружении юникодного имени
OK
Отмена
Справка
Применить

30. 

КОНТРОЛЬНІ ЗАПИТАННЯ

1.Що ви розумієте тід терміном «вимоги до інтерфейсу програмного продукту»?
2.Які ви знаєте функціональні вимоги до інтерфейсу програмного продукту?
3.Які ви знаєте нефункціональні вимоги до інтерфейсу програмного продукту?
4.На які класи діляться користувачі програмного продукту?
5.Що таке документація до програмного продукту?

ЛАБОРАТОРНА РОБОТА № 3

Створення сценарію програмного забезпечення за допомогою концепції варіантів використання

Мета роботи: виконати аналіз предметної області згідно свого варіанту та навчитися створювати сценарії варіантів використання.

Теоретичні відомості

Сценарії використання

Сценарій – це один із способів опису структури завдання. Це розповідь про виконувані дії, історія, епізод, що відбувається у даних часових рамках і у даному контексті. Різні форми сценаріїв широко застосовуються при розробці програмного забезпечення. Сценарії завдань і взаємодій в більшості випадків багаті на характеристики і володіють високою реалістичністю.

Сценарії при розробці користувацького інтерфейсу описують взаємодію між користувачем (або типом користувачів) і системою.

Звичайні сценарії мають певні обмеження при використанні їх для проектування користувацьких інтерфейсів. У них робиться наголос на реалістичність та деталі інтерфейсу, при цьому недостатньо уваги приділяється проблемам і їх загальній організації. Сценарії включають правдоподібні описи комбінацій окремих дій і завдань, тому часто буває важко виділити і зрозуміти основну суть взаємодії між компонентами.

Моделі use case

Концепція моделей use case уперше була застосована для розробки ПЗ Айваром Якобсоном як складова частина його об'єктно-орієнтованого підходу до програмної інженерії. Успіх моделі виявився наскільки значним, що з часом відбулася інтеграція елементів use case практично в усі основні методи об'єктно-орієнтованого аналізу і проектування. Незважаючи на те, що модель була розроблена спеціально для проектування об'єктно-орієнтованого ПЗ, нічого особливо «об'єктно-орієнтованого» в елементах моделі use case немає, тому їх можна застосовувати практично до усіх підходів проектування.

Елемент use case – це ситуація, варіант використання, тобто деякий випадок застосування системи. По своїй суті use case це:

- забезпечення функціональності;
- суто зовнішня точка зору (принцип «чорного ящика»);
- розповідний опис;
- опис взаємодії між користувачем (в певній ролі) і системою;
- завершене і зрозуміле користувачеві застосування системи.

Кожен елемент use case описує в розповідній формі завершену, добре визначену взаємодію, що має чітку мету з точки зору користувача. При об'єктно-орієнтованому підході елементи use case можуть описувати взаємодію з іншими системами і устаткуванням, а не

тільки із живими користувачами. Проте, коли метою є розробка призначеного для користувача інтерфейсу, можна абсолютно спокійно обмежитися розглядом тільки тих елементів use
case, які відносяться до взаємовідносин між людиною і системою.

Сутнісні елементи моделі use case

Сутнісний елемент use case – це структурований опис, виражений на мові певної предметної області та користувачів системи і спрощений, узагальнений, абстрактний, не залежний від технології і реалізації опис однієї завершеної, наповненої змістом і добре визначеної з точки зору користувачів задачі або взаємодії. Передбачається, що користувач грає певну роль по відношенню до системи, а в описі втілюються цілі і задуми взаємодії, що лежать в його основі.

Сутнісні елементи use case будуються на основі цілей і завдань користувача, а не на основі якихось конкретних механізмів або етапів, що ведуть до досягнення цих цілей. При використанні підходу, орієнтованого на зручність використання, в сутнісних елементах use case, що є структурованим описом, можна виділити три частини: виклад загальних прагнень користувача, виражених в елементах use case, плюс опис, що складається з двох частин та включає модель прагнень користувача і модель зобов'язань системи. Сутнісні елементи use case іменуються, причому за допомогою цих імен намагаються виразити призначені для користувача наміри в умовах даного варіанту використання.

Опис варіантів використання

Функціональні вимоги до системи моделюються і документуються за допомогою варіантів використання (use case). *Варіант використання* (use case)
– зв'язний елемент функціональності, що надається системою при взаємодії з дійовими особами. *Дійова особа* (actor) – роль, узагальнення елементів зовнішнього оточення системи, що поводять себе по відношенню до системи однаковим чином.

У контексті процесу управління вимогами варіанти використання трактуються наступним чином (згідно Коберна):

4. варіант використання фіксує угоду між учасниками проекту відносно поведінки системи;

5. варіант використання описує поведінку системи за різних умов, коли система відповідає на запит одного з учасників, що називається основною дійовою особою;

6. основна дійова особа ініціює взаємодію із системою, з метою досягнення певної цілі. Система відповідає, враховуючи інтереси усіх учасників.

Варіанти використання – це вид документації, який використовується у випадку, коли необхідно сконцентрувати зусилля на обговоренні принципових вимог до системи, що розробляється, а не на детальному описі системи. Стиль написання вимог залежить від масштабу, кількості учасників і критичності проекту. У загальному випадку, рекомендується

дотримуватися наступних правил:

1) назви варіантів використання мають бути діловими (нетехнічними) термінами, що мають значення для замовника;

2) кожен варіант використання має бути завершеною транзакцією між користувачем і системою, що представляє для першого певну цінність;

3) добре написаний варіант використання легко читається і складається з пропозицій, написаних в єдиній граматичній формі. Формат опису варіанту використання (згідно Коберна):

1. Ім'я – ціль у вигляді короткої дієслівної фрази.
2. Контекст використання – розширений опис цілі.
3. Область дії – ідентифікує розглядувану систему.
4. Рівень – визначає рівень цілі.
5. Основна дійова особа.
6. Інші учасники і їх інтереси.
7. Передумова (визначає, виконання якої умови гарантує система перед тим, як дозволити запуск варіанту використання).
8. Мінімальні гарантії (найменші обіцянки системи учасникам, зокрема у випадку, коли ціль основної дійової особи не може бути досягнута).
9. Гарантії успіху або постумова (встановлює, що інтереси учасників задовольняються після успішного завершення варіанту використання у кінці основного сценарію).
10. Тригер (подія, яка запускає варіант використання).
11. Основний сценарій або потік (простий для розуміння типовий сценарій, в якому досягається ціль основної дійової особи і задовольняються інтереси усіх учасників). Кожен крок основного сценарію описує: взаємодію двох дійових осіб ("Клієнт вводить адресу"); крок підтвердження для захисту інтересів учасника ("Система підтверджує PIN-код"); внутрішня зміна для задоволення інтересу учасника ("Система виводить суму з балансу").
12. Розширення (запускаються при виникненні певної умови та містять послідовність кроків, які описують, що відбувається за цієї умови).
13. Допоміжна інформація.

Шаблон RUP для опису варіанту використання схожий на метод запропонований Коберном. Він відрізняється лише тим, що альтернативні потоки описуються окремо від інших розширень.

Побудову варіантів використання засобами Axure RP можна здійснити використавши Панель віджетів і вибравши з перелічених умов Select Library: Common, Flow. Основні елементи для побудови варіантів використання знаходяться саме в розділі Flow. Додаткові елементи (лінії, заголовки, мітки) використовуйте з розділу Common.

Завдання на лабораторну роботу:

1. Ознайомитися з теоретичними відомостями по роботі.
2. Провести аналіз предметної області відповідно до свого варіанту.
3. Скласти 5 сценаріїв варіантів використання програмного забезпечення користувачем згідно з форматом опису Коберна.
4. Зробити висновки по роботі.

Варіанти завдань

1. Інтернет-магазин. Мають бути реалізовані сценарії: купівля товару, пошук товару, додавання нового товару у базу даних магазину, перегляд і обробка замовлень покупців, реєстрація нового покупця.

2. Книжковий каталог. Мають бути реалізовані сценарії: додавання нової книги, пошук книги за декількома полями, бронювання книги, списання старих книг, реєстрація користувачів каталогу.

3. Адресна книга. Мають бути реалізовані сценарії: додавання нового абонента, додавання категорій абонентів, пошук абонентів за декількома полями, додавання адміністраторів каталогу (користувачів, які мають право редагувати дані адресної книги), редагування даних абонента.

4. Розклад занять. Мають бути реалізовані сценарії: додавання нової групи, додавання занять (з вказанням назви предмета, часу, аудиторії, групи, тижня, викладача, типу заняття), перегляд списку занять по вибраній даті, додавання списку викладачів, пошук занять за декількома полями (предмет, викладач, група, час, тип заняття).

5. База студентів. Мають бути реалізовані сценарії: додавання нової групи, додавання нового студента, пошук студента по різних полях, додавання інформації про оцінки по різних предметах, відрахування студента.

6. Прайс-лист фірми. Мають бути реалізовані сценарії: додавання нової категорії товарів, додавання нового товару, пошук товару по різних полях, додавання адміністратора прайс-листа (користувачів, які мають право редагувати прайс-лист), переміщення товару з однієї категорії в іншу.

7. **База складу фірми.** Мають бути реалізовані сценарії: додавання нового товару на склад, списання товару, видача товару, пошук товару по різних полях, зміна місцерозташування товару на складі.

8. **Аптечна база.** Мають бути реалізовані сценарії: прийом замовлення від клієнта на виготовлення розчину, продаж ліків, списання прострочених ліків, додавання нових ліків у базу даних, пошук замовлень по різних полях.

9. **Ювелірний завод.** Мають бути реалізовані сценарії: прийом замовлення від клієнта на виготовлення виробу, продаж ювелірного виробу, обмін власного виробу на один із заводських виробів, додавання нової проби виробу у базу даних, пошук товару за різними

ознаками (срібло золото, брильянти).

10. **Автомийка.** Мають бути реалізовані сценарії: прийом клієнта на автомийку, бронювання мийки на задану годину, вибір послуг, додавання хімічних матеріалів, якими робляться послуги клієнтів у базу даних.

11. **Рослинний магазин.** Мають бути реалізовані сценарії: прийом замовлення від клієнта на замовлення рослин, модифікація рослин, додавання нових рослин та окремо видів у базу даних, пошук видів рослин по різних полях.

12. **Продуктовий склад.** Мають бути реалізовані сценарії: прийом замовлення від торгових марок, продаж продуктів по собівартості, списання прострочених продуктів, додавання нових продуктів у базу даних, пошук продуктів за різними ознаками (задаєте самостійно).

13. **Футбольна команда.** Мають бути реалізовані сценарії: дані про футболістів, трансферне вікно команди, розклад ігор команди, додавання футболістів з молодіжної команди у основну базу даних,участь команди у турнірах, турнірні таблиці з результатами участі у різних турнірах.

Приклад виконання

1. **Ім'я**

   Реєстрація учасника.

2. **Контекст використання**

   Реєстрація учасника на веб-сайті інтернет-аукціону з метою подальшої участі у торгах.

3. **Область дії**

   Модуль реєстрації учасників інтернет-аукціону.

4. **Рівень**

   Ціль користувача

5. **Основна дійова особа**

   Учасник – бере участь в інтернет-аукціоні.

6. **Інші учасники і їх інтереси**

   Адміністратор – обслуговує веб-сайт інтернет-аукціону.

7. **Передумова**

   Учасник повинен бути повнолітнім.

8. **Мінімальні гарантії**

   Наявність мінімальної реєстраційної інформації для пошуку варіанту вирішення проблеми, що виникла в процесі реєстрації.

9. **Гарантії успіху або постумова**

   Успішна реєстрація на веб-сайті інтернет-аукціону.

10. **Тригер**

Перехід на веб-сторінку з реєстраційною формою.

11.Основний сценарій або потік

1) учасник вводить особисті дані, що вимагаються системою для реєстрації;
2) учасник погоджується з умовами використання вибраного сервісу;
3) система виконує процедуру перевірки коректності введених даних;
4) система відправляє повідомлення на електронну пошту, що вказав учасник під час реєстрації;
5) учасник підтверджує адресу своєї електронної пошти шляхом переходу по гіперпосиланню, що було згенероване системою.

12.Розширення

Учасник ввів некоректні персональні дані:

- виводиться повідомлення про те, що були введені некоректні дані. Учасник не заповнив обов'язкові поля веб-форми:
- виводиться повідомлення про те, що не всі обов'язкові поля веб-форми було заповнено.

13.Допоміжна інформація

Веб-сторінка з умовами використання вибраного сервісу.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке діаграми використання?
2. З чого складаються діаграми використання?
3. Як побудувати діаграму використання засобами програми Axure RT?
4. Що таке діаграми діяльності?

ЛАБОРАТОРНА РОБОТА № 4

Побудова use case діаграм і діаграм діяльності

Мета роботи: виконати аналіз предметної області згідно свого варіанту та навчитися виконувати побудову use case діаграм та діаграм діяльності.

Теоретичні відомості

Карти елементів use case

Елементи use case не існують окремо від зовнішнього світу. Повноцінна програмна система повинна забезпечувати підтримку десятків, а то і сотень елементів use case, причому, всередині певним чином пов'язаного з нею програмного додатку повинен існувати взаємозв'язок між цими елементами. Відображення взаємозв'язку між програмними додатками дає можливість описати загальну структуру завдання, що вирішується програмним додатком і його інтерфейсом. Карта елементів use case для цього завдання розбиває усі функціональні можливості системи на множину взаємозв'язаних сутнісних елементів use case. Виділивши усі важливі взаємодії і показавши взаємодію між ними, можна створити спрощену загальну модель завдань, що вирішуються системою, і можливостей, які вона зобов'язана надавати.

Повноцінна модель use case є множиною описів, що визначають суть усіх елементів use case, і карту цих елементів, що представляє взаємозв’язки між ними. Між сутнісними елементами use case можуть існувати взаємозв’язки різних типів, включаючи спеціалізацію, розширення, композицію, а також схожість. Знання про ці стосунки дозволяє аналітикові або розробникові виділити загальні елементи завдання і в результаті створити спрощену його модель.

Спеціалізація

Деякі елементи use case можуть бути спеціалізованими версіями інших елементів. Наприклад, при розробці додатка «банкомат» елементи use case «Отримання грошей», «Розміщення вкладу» і «Запит стану» є підкласами, або спеціалізованими варіантами абстрактного класу взаємодій, який може бути названий «Використання банкомату». Що стосується відношення між елементами «Отримання грошей» і «Використання банкомату», то його можна охарактеризувати як класифікацію, або спеціалізацію. Такий тип відношення означає, що один елемент use case «є» («is-a») спеціалізацією іншого. У об'єктно-орієнтованому аналізі і проектуванні таке відношення відповідає відношенню клас-підклас.

Спеціалізація дає можливість спростити загальну модель use case шляхом відділення загальних або універсальних форм взаємодії від специфічних форм, адаптованих для вужчого застосування. Таким чином, немає необхідності переписувати наново загальні патерни

проектування. Достатньо описати їх один раз, а потім лише «повторно використовувати» («reuse»), посилаючись на них. Як видно з рис. 1, для відображення відношення спеціалізації використовується подвійна стрілка. Поряд із стрілкою можна зустріти підпис «is-a» або «specialize», залежно від контексту.

Рисунок 1 – Відношення спеціалізації на карті елементів use case

Розширення

Однією із інновацій в об'єктно-орієнтованій програмній інженерії, навіяних ідеями Якобсона, стало визнання розширення одним з можливих стосунків між елементами use case. Говорять, що один елемент «розширює» інший, коли він містить альтернативні патерни взаємодії, які увійдуть до розширюваного елементу. Наприклад, у випадку елементу use case, призначеного для зміни зовнішнього вигляду певної частини екрану користувачу необхідно виконати пошук у системі файлу, що містить потрібну картинку або значок.

Розширення – це вдала концепція, що дозволяє значно спростити сутнісні моделі use case. На карті елементів use case відношення розширення зображається у вигляді пунктирної лінії із стрілкою і має підпис «extend», як показано на рис. 2. Якщо для розширення необхідний додатковий опис, в нього може бути включена примітка, що показує, які елементи use case розширюються.

Рисунок 2 – Відношення розширення на карті елементів use case

Композиція

Елементи use case можна розкласти на складові частини, або піделементи, що є

підпорядкованими або включеними патернами взаємодії. Відношення композиції позначається на карті елементів use case пунктирною стрілкою, що вказує на піделемент use case і має мітку «include» як показано на рис. 3.

Взаємодія, що описується суперелементом use case, здійснюється за допомогою взаємодій, що входять в піделемент або піделементи, причому опис

суперелементу посилатиметься на усі використовувані піделементи. Наприклад, елемент use case під назвою «Початок протоколювання задачі», створений для програми, що відстежує хід виконання задач, може використати елементи «Авторизація доступу» і «Ввід параметрів задачі». Такий спосіб моделювання взаємодій дозволяє розділити незалежні і майже ніяк не пов'язані між собою підзадачі «Авторизація доступу» ту «Ввід параметрів задачі».

Рисунок 3 – Відношення композиції на карті елементів use case

Діаграми діяльності

Для опису функціональних вимог окрім діаграм варіантів використання використовуються також діаграми діяльності. Вони застосовуються для:

- опису поведінки, що включає велику кількість паралельних процесів;
- аналізу варіанту використання (описують послідовність дій і їх взаємозв'язок);
- аналізу потоків робіт (workflow) в різних варіантах використання.

Коли варіанти використання взаємодіють один з одним, діаграми діяльності є засобом представлення і аналізу їх поведінки. Приклад діаграми діяльності представлений на рис. 4.

Рисунок 4 – Приклад діаграми діяльності

Для побудови діаграм варіантів використання використовуємо розділи: Common та Flow з панелі віджетів. Для роботи нам знадобляться такі елементи як: Actor, Ellipse, Rounded Rectangle, Horizontal Line, Vertical Line. Для більшточного відображення і розробки вищезгаданого завдання потрібно скористатися властивостями відповідних елементів.

Завдання на лабораторну роботу

1.Ознайомитися з теоретичними відомостями по роботі.

2.Провести аналіз предметної області відповідно до свого варіанту.

3.Побудувати use case діаграми для кожного варіанту використання.

4.Побудувати діаграми діяльності для кожного варіанту використання.

5.Зробити висновки по роботі.

Варіанти завдань

1.Інтернет-магазин. Мають бути реалізовані сценарії: купівля товару, пошук товару, додавання нового товару у базу даних магазину, перегляд і обробка замовлень покупців, реєстрація нового покупця.

2.Книжковий каталог. Мають бути реалізовані сценарії: додавання нової книги, пошук книги за декількома полями, бронювання книги, списання старих книг, реєстрація користувачів каталогу.

3Адресна книга. Мають бути реалізовані сценарії: додавання нового абонента, додавання категорій абонентів, пошук абонентів за декількома полями, додавання адміністраторів каталогу (користувачів, які мають право редагувати дані адресної книги), редагування даних абонента.

4.Розклад занять. Мають бути реалізовані сценарії: додавання нової групи, додавання занять (з вказанням назви предмета, часу, аудиторії, групи, тижня, викладача, типу заняття), перегляд списку занять по вибраній даті, додавання списку викладачів, пошук занять за декількома полями (предмет, викладач, група, час, тип заняття).

5. База студентів. Мають бути реалізовані сценарії: додавання нової групи, додавання нового студента, пошук студента по різних полях, додавання інформації про

оцінки по різних предметах, відрахування студента.

6. Прайс-лист фірми. Мають бути реалізовані сценарії: додавання нової категорії товарів, додавання нового товару, пошук товару по різних полях, додавання адміністратора прайс-листа (користувачів, які мають право редагувати прайс-лист), переміщення товару з однієї категорії в іншу.

7.8База складу фірми. Мають бути реалізовані сценарії: додавання нового товару на склад, списання товару, видача товару, пошук товару по різних полях, зміна місцерозташування товару на складі.

8.Аптечна база. Мають бути реалізовані сценарії: прийом замовлення від клієнта на виготовлення розчину, продаж ліків, списання прострочених ліків, додавання нових ліків у базу даних, пошук замовлень по різних полях.

9.Ювелірний завод. Мають бути реалізовані сценарії: прийом замовлення від клієнта на виготовлення виробу, продаж ювелірного виробу, обмін власного виробу на один із заводських виробів, додавання нової проби виробу у базу даних, пошук товару за різними ознаками (срібло золото, брильянти).

10.Автомийка. Мають бути реалізовані сценарії: прийом клієнта на автомийку, бронювання мийки на задану годину, вибір послуг, додавання хімічних матеріалів, якими робляться послуги клієнтів у базу даних.

11.Рослинний магазин. Мають бути реалізовані сценарії: прийом замовлення від клієнта на замовлення рослин, модифікація рослин, додавання нових рослин та окремо видів у базу даних, пошук видів рослин по різних полях.

12.Продуктовий склад. Мають бути реалізовані сценарії: прийом замовлення від торгових марок, продаж продуктів по собівартості, списання прострочених продуктів, додавання нових продуктів у базу даних, пошук продуктів за різними ознаками (задаєте самостійно).

13.Футбольна команда. Мають бути реалізовані сценарії: дані про футболістів, трансферне вікно команди, розклад ігор команди, додавання футболістів з молодіжної команди у основну базу даних,участь команди у турнірах, турнірні таблиці з результатами участі у різних турнірах.

КОНТРОЛЬНІ ЗАПИТАННЯ

1.Хто такі актори в діаграмах використання ?

2.Як позначають актори на діаграмах?

3.Для чого використовуються діаграми діяльності?

4. У чому сутність діаграми використання?

5. Як можна видалити діючі особи з діаграми варіантів використання?

ЛАБОРАТОРНА РОБОТА № 5

Створення прототипу інтерфейсу Windows-додатку

Мета роботи: виконати аналіз предметної області згідно свого варіанту та навчитися створювати прототипи інтерфейсів за допомогою програми Axure RP.

Теоретичні відомості

Axure як інструмент візуального проектування

Останнім часом розробники веб-проектів все частіше починають замислюватися над тим, що до початку розробки продукту було б непогано дізнатися, що це буде за продукт і як він буде виглядати.

Ця тенденція не може не радувати як замовників, так і самих розробників - адже процес проектування дозволяє знищити всі найбільш проблемні моменти ще до початку перетворення абстрактного технічного завдання в кінцевий продукт.

Коштів для візуального проектування стає все більше. Але все більше шанувальників з'являється у досить важкій і функціональної програми Axure RP Pro.

Axure є спеціальним інструментом для прототипування веб-сайтів та веб-додатків. Axure увібрала в себе все краще, що вміє Visio в області розробки концепції та схем, а також все найчарівніше з дизайнерських програм, що дозволило їй стати новим інструментом, який стоїть на стику проектування і дизайну - цих двох таких близьких, але зовсім різних світів.

Основні переваги Axure

1. *Drag and drop widgets.* Можливість "взяти" мишкою будь потрібний елемент інтерфейсу з колекції Axure і помістити його в прототип в будь-яке потрібне вам місце без написання коду

2. *Генерація функціональних (чи інтерактивних) прототипів*. Одним рухом руки рисунок перетворюється в файл html, який можна відкрити в браузері.

3. *Економія часу на внесення змін і виправлень.* У Axure можна внести зміни в усі сторінки прототипів одночасно, просто підправивши шаблон

4. *Симулювати RIA (складні інтерактивні елементи).* Як імітувати AJAX без AJAX? За допомогою нескладної гри з шарами в Axure.

Ta-dam. Axure може сама автоматично згенерувати специфікацію інтерфейсів.

При створенні програмних інтерфейсів рекомендується використовувати існуючі принципи проектування інтерфейсів користувача. Далі наводяться шість принципів, що увібрали в себе знання про розробку ефективних інтерфейсів користувача. Кожен з них включає декілька пов'язаних між собою ідей, більш деталізованих в порівнянні із загальними питаннями. Цими загальними питаннями є структура, простота, видимість, зворотний зв'язок, толерантність і повторне використання.

Структурний принцип

Організація інтерфейсу користувача має бути доцільною, осмисленою і зручною. Вона повинна базуватися на чітких, цілісних моделях, очевидних і зрозумілих

користувачам. При цьому споріднені поняття мають бути пов'язані, а незалежні – розділені. Несхожі елементи повинні диференціюватися, а схожі – виглядати схоже.

Структурний принцип пов'язаний із загальною архітектурою інтерфейсу і безпосередньо відображає уявлення про інтерфейс користувача як про діалог між розробниками і користувачами. Організація хороших інтерфейсів продумується дуже ретельно, так, щоб відображати структуру вирішуваних системою задач і спосіб мислення користувачів відносно цих задач. Дуже часто, особливо при використанні сучасних візуальних середовищ розробки, розташування візуальних компонентів усередині форм або діалогів і їх розподіл між ними виявляється майже випадковим і відбиває у кращому разі послідовність, в якій програмістами розглядалися ті або інші питання. Властивості і функції, які найчастіше використовуються спільно або розглядаються користувачами як пов'язані одна з одною, повинні розташовуватися разом або, принаймні, мають бути чітко і ясно взаємозв'язані. Що ж до тих елементів, які в контексті завдання або у свідомості користувача ніяк не пов'язані між собою, то вони мають бути рознесені в інтерфейсі. Подібне має бути подібне. Схожа інформація має бути організована за допомогою схожих рішень, а об'єкти, що мають схожу поведінку, повинні мати загальне представлення.

Принцип простоти

Слід максимально спрощувати керування найбільш поширеними операціями. При цьому спілкування з користувачем повинне вестися на зрозумілій для нього мові. Повинні надаватися посилання, що логічним чином вказують на складніші процедури.

Процес проектування інтерфейсу – це завжди боротьба за компроміс. Спрощення чогось одного неминуче призводить до ускладнення чогось іншого. Якщо зменшити кількість меню, збільшиться число пунктів в кожному з них. Якщо зробити маленькими усі діалогові вікна, включивши в них якомога менше елементів, будь-яка взаємодія користувача з системою обернеться для нього необхідністю звертатися до великої кількості таких вікон.

Неможливо зробити усе на світі простим. Наслідування принципу простоти вимагає знання того, які задачі виконуються користувачем найчастіше і які з них, з точки зору користувача є простішими. Саме такі задачі слід спрощувати, щоб користувач міг швидко їх вирішити.

Принцип видимості

Усі функції і дані, необхідні для виконання конкретної задачі, мають бути на виду, щоб користувач не відволікався на додаткову і надмірну інформацію.

Принцип видимості пов'язаний з проектуванням таких інтерфейсів користувача, в яких видно усі елементи, потрібні для виконання поставленої задачі. Мета – перейти від філософії WYSIWYG (What You See Is What You Get – що бачиш на екрані, то і отримаєш в результаті) до філософії WYSIWYN (What You See Is What You Need – на екрані бачиш те, що тобі потрібно). Інтерфейси WYSIWYN залишають видимими ті, і тільки ті елементи, які дійсно потрібні користувачеві для виконання операції.

З одного боку, в задачу проектування входить створення такого інтерфейсу, в якому були б явно виділені усі потрібні і важливі функції. З іншого боку, хороший інтерфейс не повинен завалювати користувача занадто великою кількістю можливих варіантів або бентежити його надмірною інформацією. WYSIWYN-інтерфейси є кращі уже тим, що вони беруть до уваги обмеженість об'єму «оперативної пам'яті» людини і здатність дізнаватися речі швидше, ніж згадувати. Навантаження на довготривалу пам'ять зменшується за рахунок того, що користувач постійно бачить усі необхідні опції і варіанти. На короткочасну пам'ять навантаження знижується за рахунок того, що користувачу не доводиться запам'ятовувати і потім відтворювати інформацію, що міститься в будь-якій іншій частині інтерфейсу.

Принцип зворотного зв'язку

Повідомляйте користувачів про дії системи, її реакції, зміни стану або ситуації, про помилки і винятки, які важливі для них. Повідомлення мають бути чіткими, короткими, однозначними і написаними на мові, зрозумілій користувачу.

Хороші інтерфейси користувача знаходяться в діалозі з користувачами, повідомляючи їх про те, що відбувається в системі. Принцип зворотного зв'язку вказує розробникам деякі правила цього діалогу.

Практичні системи інформують користувача про велику кількість речей. Наприклад, вони повинні надавати йому можливість дізнатися про те, як сприймаються введені ним дані. Кожного разу, коли міняється внутрішній стан системи, і коли це має будь-який вплив на роботу користувача, його слід повідомляти про це, особливо якщо міняється інтерпретація системою його дій. Зрозуміло, користувач повинен знати про дії, які заборонені або ігноруються. При цьому принцип зворотного зв'язку не може служити виправданням створення нескінченної кількості вікон з повідомленнями. Інформування користувача – не самоціль, а спосіб організації діалогу в компактній і природній формі.

Користувачам також потрібно повідомляти про помилки і виняткові ситуації. У багатьох програмах ці повідомлення, на жаль, неінформативні і здатні ввести в оману. Можна іноді зустріти навіть ображаючі повідомлення, після прочитання яких користувачеві може стати ніяково. Навряд чи людина надихнеться, скажімо, таким повідомленням: «Неправильно! Введіть коректні дані!». Таке повідомлення не лише неявно припускає, що користувач – некомпетентна людина, але і, по суті справи, не надає ніякій інформації. Тут не сказано, що саме неправильно і чому.

Грамотно складені повідомлення про помилки – це ще один приклад хорошої організації спілкування з користувачем. Рекомендації тут можна дати такі: стислість; мова, зрозуміла користувачеві; простота розуміння. Передусім, інформативним має бути заголовок повідомлення. Він повинен в стислій формі описувати проблему, а вже саме
повідомлення повинне розкривати подробиці і пропонувати способи рішення або послідовність коригуючих дій.

Принцип толерантності

Інтерфейс має бути гнучким і толерантним. Шкоду, що наноситься помилками користувачів, необхідно знижувати за рахунок можливості відміни і повтору дій, а також за рахунок запобігання появі цих помилок шляхом аналізу різних форматів введення і розумної інтерпретації будь-яких дій користувача.

Інтерфейс можна робити більш або менш толерантним залежно від того, які дані перевіряються і коли. Перевірка за один раз усіх полів після закінчення введення даних – практика поширена і іноді виправдана. При цьому толерантності системі додасть автоматичне підсвічування поля з неправильними даними, установка на ньому курсору, плюс коротке, інформативне повідомлення в рядку стану. Найбільше користувачі страждають від програм, які після закінчення введення даних в усі поля перевіряють форму, і у разі помилки хоча б в одному з полів залишають користувача знову наодинці з порожнім бланком. Здавалося б, таке рішення страшенно безглузде, але воно зустрічається дуже часто, у тому числі в комерційних програмах.

У цілому перевірка невживаних полів або полів, які ніяк не обробляються системою і представляють інтерес тільки для користувачів (у тому вигляді, в якому вони були введені), веде до зниження толерантності програмного забезпечення. Наприклад, перевірка того, що в полі приміток є присутніми тільки буквено-цифрові символи, надмірна, оскільки у випадку коли користувач захоче ввести у це поле спецсимвол, то у нього виникнуть проблеми.

Принцип повторного використання

Слід багаторазово використовувати внутрішні та зовнішні компоненти і принципи поведінки системи, підтримуючи стійкість осмислено, а не просто за рахунок надмірності. Це сприяє зменшенню об'єму інформації, яку користувачам доводиться запам'ятовувати і про яку доводиться думати кожного разу наново.

Застосовуючи повторне використання зовнішніх і внутрішніх компонентів та рішень, що поширюються на усю систему, розробник може створити не лише більш цілісний інтерфейс, але і в результаті здешевити програмний продукт. Прагнення до однієї лише стійкості підвищує вартість розробки. Треба прагнути до стійкості в контексті вирішуваних системою задач і області її використання. Стійкість жодним чином не може бути самоціллю, інакше вона може виглядати дещо дурнувато і навіть приводити, як не дивно, до невдалих з точки зору несуперечності систем.

Багато прийнятих стандартів і загальновизнані компоненти інтерфейсу користувача являють собою приклади невдалих спроб реалізації стійкості. Стандартні програмні платформи іноді нав'язують розробникові погано продумані і невдало спроектовані рішення. Звичайні діалогові вікна, скажімо, можуть забезпечувати цілісність і при цьому бути дуже низькосортними, вибір стандартних гарячих клавіш виявляється випадковим і таким, що аж ніяк не відповідає принципу стійкості.

Завдання на лабораторну роботу

1.Ознайомитися з теоретичними відомостями по роботі.

2.Вивчити середовище програми Axure RP.

3.Розробити графічний прототип інтерфейсу windows-додатку відповідно до основних принципів проектування інтерфейсу. Інтерфейс має бути достатній для виконання усіх сценаріїв.

4.Зробити висновки по роботі.

КОНТРОЛЬНІ ЗАПИТАННЯ

1.Що таке принцип простоти?
2.Що означає структурний принцип?
3.Що таке принцип видимості?
4..Що таке принцип зворотнього звязку?
5.Що означає принцип повторного використання?

ЛАБОРАТОРНА РОБОТА № 6

Проектування графічних інтерфейсів засобами Axure RP

Мета роботи: ознайомитись із середовищем проектування і документування графічних інтерфейсів Axure RP. Отримати навички по роботі із основними інструментами Axure RP. Навчитися створювати звичайні гіпертекстові посилання та використовувати множинні сценарії однієї взаємодії.

Теоретичні відомості

Axure RP – це інструмент для проектування і документування графічних інтерфейсів, який використовують UX-спеціалісти, інформаційні архітектори, бізнес-аналітики та менеджери з виробництва для створення макетів з примітками, інтерактивних прототипів і функціональних специфікацій настільних програм та веб-сайтів.

6.1 Карта інтерфейсу

У центральній частині програми Axure RP розташовується вікно створення макетів (Wireframe Pane), що підтримує перетягування елементів (Widgets) для проектування макету. Навколо вікна створення розмітки розташовані панелі інструментів, які можна переміщати або приховувати. За допомогою меню «Вид» (View) у верхньому меню можна приховувати або відображати різноманітні панелі (рис. 6.1).

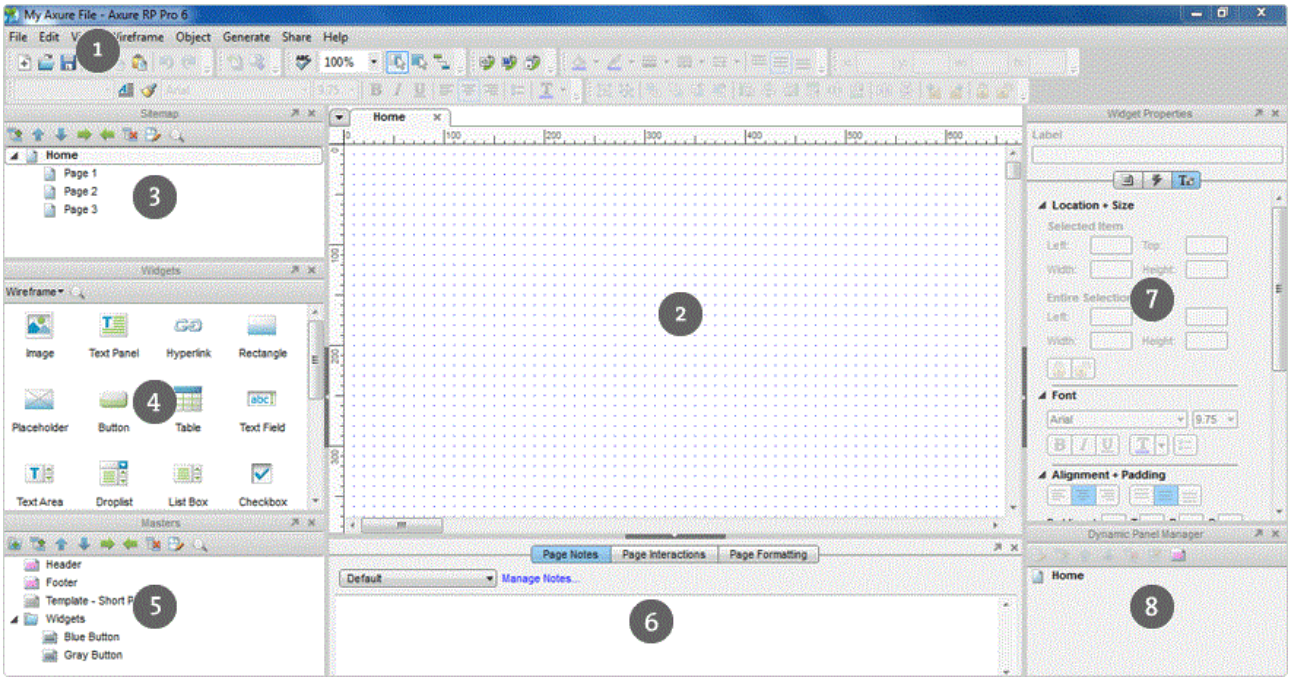


Рисунок 6.1 – Карта інтерфейсу Axure RP

Карта інтерфейсу Axure RP включає наступні компоненти:

1. **Головне меню і панель інструментів** (Main Menu і Toolbar) призначені для виконання загальних завдань на зразок відкриття і збереження файлів, а також для форматування елементів на сторінці. Можливе перемикання між режимами виділення об'єктів (виділення зачеплених об'єктів і виділення повністю обведених об'єктів), а також

перемикання в режим з'єднання.

2. **Вікно створення макету** (Wireframe Pane) дозволяє проектувати сторінки в просторі з можливістю перетягування елементів і додатковими опціями на зразок прив'язки до сітки та направляючих.

3. **Карта проекту** (Sitemap Pane) призначена для додавання, видалення, перейменування і керування сторінками проекту.

4. **Панель віджетів** (Widgets Pane) – це набір готових елементів для дизайну сторінок: кнопки, зображення, текстові блоки і форми. Елементи можна перетягувати мишкою у вікно створення макету. Є також можливим створювати свої власні панелі або завантажувати готові бібліотеки.

5. **Панель майстрів** (Masters pane) дозволяє створювати, видаляти, перейменовувати і налаштовувати майстри з можливістю їх подальшого використання в дизайні.

6. **Вікно налаштувань сторінки** (Page Properties Pane) призначене для додавання і редагування заміток і взаємодій рівнів сторінки, а також для форматування сторінок дизайну.

7. **Вікно налаштування елементів** (Widget Properties Pane) дозволяє редагувати примітки, взаємодії елементів і здійснювати їх форматування.
   - **примітки** (Annotations) – для додавання і налаштування заміток і опису елементів.
   - **взаємодії** (Interactions) – для створення взаємодій між сторінками за допомогою посилань на інші сторінки, спливаючих вікон, а також елементів, що динамічно з'являються і зникають.
   - **форматування** (Formatting) – для редагування таких стилів і властивостей елементів, як розмір, положення на сторінці, шрифти, вирівнювання і відступи тексту, а також графічний стиль.

8. **Вікно керування динамічними панелями** (Dynamic Panel Manager Pane) дозволяє приховувати і відображати динамічні панелі у вікні створення розмітки, а також додавати, видаляти і налаштовувати динамічні панелі.

6.2 Карта проекту

Карта проекту застосовується для додавання, видалення і систематизації сторінок в дизайні. Кількість сторінок, що додаються, нічим не обмежена, проте якщо проект складається з декількох сотень сторінок, то краще розбити його на декілька файлів.

Додавання і видалення сторінок

Щоб додати сторінку, натисніть кнопку «Add Child Page» (Додати дочірню сторінку) на панелі інструментів «Карта проекту» (рис. 6.2).

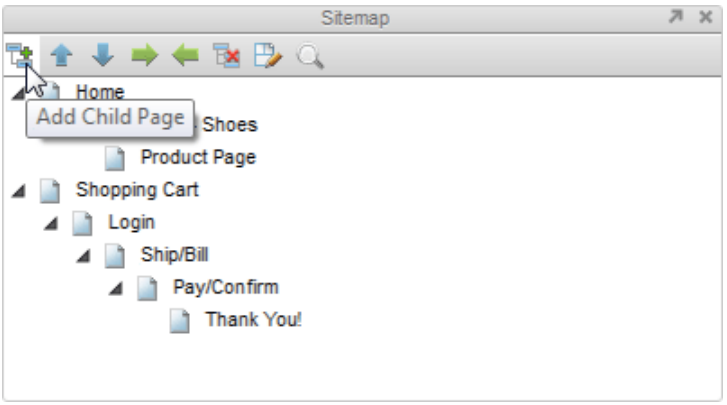


Рисунок 6.2 – Додавання нової сторінки

Щоб перейменувати вибрану сторінку, клікніть двічі по її назві. Перейменування сторінок в структурі ніяк не впливає на зв'язки між ними в прототипі.

Щоб видалити сторінку, виберіть її і натисніть кнопку «Delete Page» (Видалити сторінку) на панелі «Карта проекту».

Також додавати, перейменовувати і видаляти сторінки можна за допомогою контекстного меню, яке з'являється при натисканні правої кнопки миші на назві сторінки, або при натисканні стрілки вниз біля назви сторінки.

Систематизація сторінок

Для того, щоб упорядкувати сторінки в карті проекту, їх можна перетягувати мишкою або переміщати за допомогою іконок із стрілками, розміщених на панелі інструментів (рис. 6.3). Переміщення сторінок не впливає на взаємодію між ними.

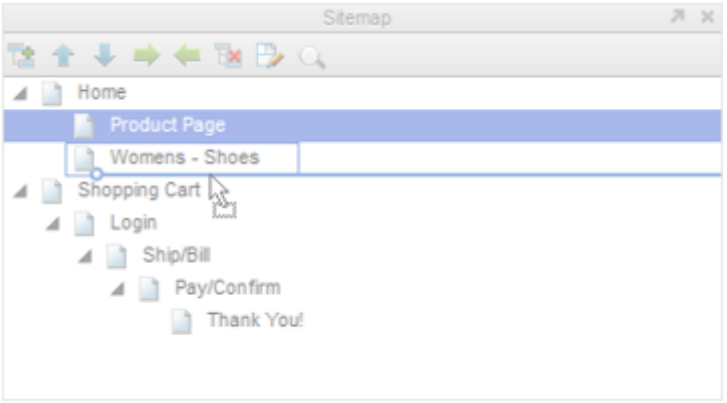


Рисунок 6.3 – Упорядкування сторінок у карті проекту

Редагування дизайну сторінки

Щоб відредагувати дизайн сторінки, подвійним клацанням відкрийте її у вікні створення макету.

Для зручного доступу вже відкриті сторінки розташовані у вигляді вкладок над вікном створення макету. Щоб змінити порядок вкладок, їх можна перетягувати мишкою. Щоб подивитися список усіх відкритих вкладок або скористатися функцією «Закрити усі вкладки» (Close All Tabs) чи «Закрити усі вкладки окрім поточної» (Close Other Tabs), натисніть на стрілку вниз біля панелі вкладок.

6.3 Віджети

Віджети (Widgets) – це елементи інтерфейсу, що використовуються для створення макетів. Панель віджетів включає бібліотеку найбільш популярних елементів, таких як кнопки, зображення та текстові блоки (рис. 6.4).

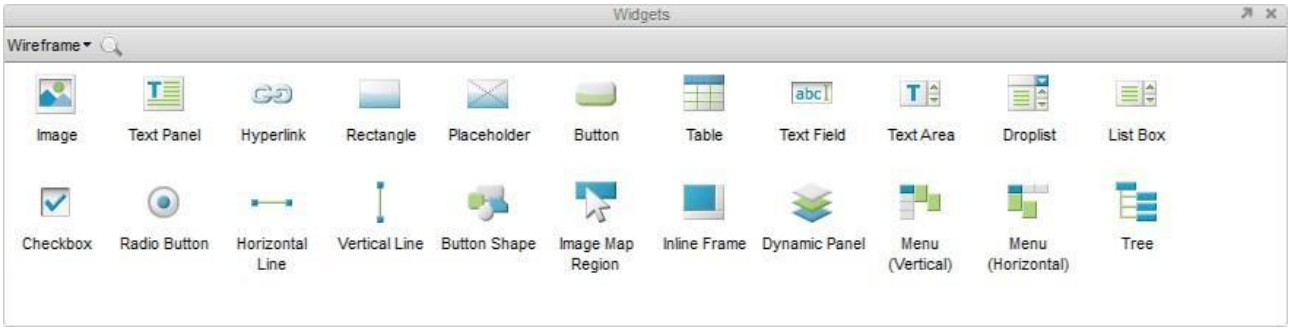


Рисунок 6.4 – Бібліотека стандартних віджетів

Окрім стандартного набору віджетів, можна створювати свої власні або закачувати і застосовувати доступні бібліотеки елементів інтерфейсу.

Вибір і пошук бібліотек віджетів

Щоб вибрати бібліотеку елементів, відкрийте випадне меню і виберіть необхідну із списку бібліотеку (рис. 6.5). На панелі віджетів з'являться елементи інтерфейсу, які входять в цю бібліотеку. Щоб подивитися усі елементи в усіх завантажених бібліотеках, натисніть «Усі бібліотеки» (All Libraries).

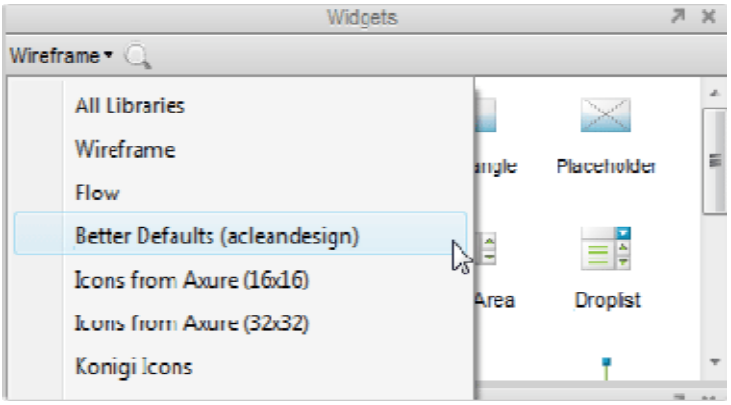


Рисунок 6.5 – Вибір бібліотеки елементів

Щоб виконати пошук по поточній бібліотеці, натисніть на значок пошуку і введіть текст в поле.

Додавання, переміщення і зміна розмірів віджетів

Щоб додати віджет у вікно створення макету, перетягніть його з панелі бібліотеки віджетів (рис. 6.6). Також можна скопіювати і вставити елементи з однієї сторінки на іншу.

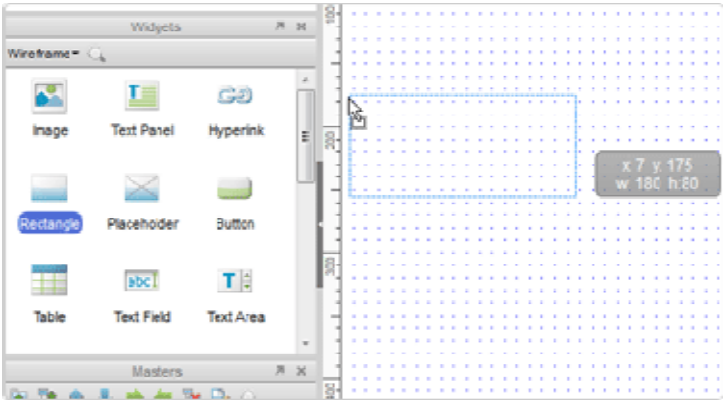


Рисунок 6.6 – Додавання віджету на сторінку

Щоб перемістити виджет, перетягніть його мишкою на потрібне місце або посуньте за допомогою кнопок переміщення на клавіатурі.

Щоб змінити розміри елементу, мишкою потягніть його рамку. Щоб вказати точне розташування і розмір віджету, використайте функцію «Розміщення і розміри» (Location and size), яка розташована на панелі інструментів, а також на панелі властивостей віджетів.

Також можна вибрати відразу декілька елементів і одночасно змінювати їх розмір і розміщення.

Редагування стилів і параметрів віджетів

Редагувати стилі і параметри віджетів (рис. 6.7) можна декількома способами:

1. **Подвійне клацання.** Подвійне клацання по віджету дозволяє редагувати основні параметри конкретного елементу. Наприклад, подвійне клацання по віджету «Зображення» (Image) відкриває діалогове вікно для імпортування зображення, а подвійне клацання по випадаючому списку відкриває діалогове вікно додавання пунктів в цей список.
2. **Панель інструментів.** За допомогою кнопок на панелі інструментів, яка розташована прямо над вікном створення макету, можна змінювати такі стилі віджетів, як: назва шрифту, його розмір і тип, колір заливки, колір границь, а також розміри і розміщення елементів. Також можна вибрати відразу декілька елементів і застосувати до них такі інструменти як угрупування, порядок, вирівнювання та рівномірний розподіл.
3. **Вкладка Форматування у вікні властивостей віджетів.** Дозволяє змінювати стиль і властивості елементів. Включає функції налаштування шрифту, вирівнювання, відступу, стилю, порядку, заливки, ліній, границь, а також розміщення і розміру елементів.
4. **Контекстне меню.** Контекстне меню викликається клацанням правої кнопки миші по потрібному елементу і включає додаткові функції налаштування віджетів. Перелік функцій залежить від типу віджета

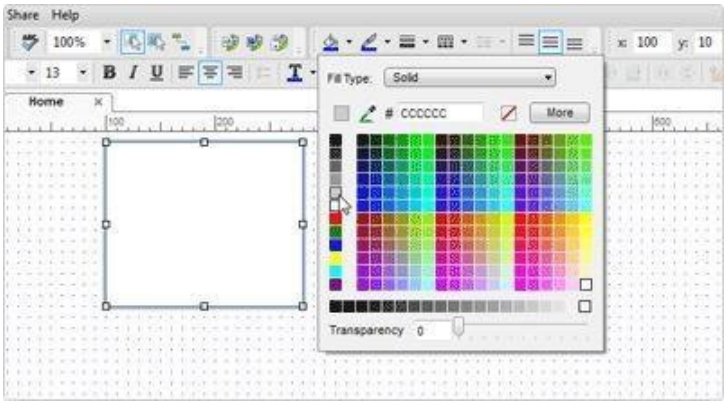


Рисунок 6.7 – Редагування стилів і параметрів віджета

Робота з декількома елементами одночасно

Вибравши відразу декілька елементів, і викликавши контекстне меню правою кнопкою миші, можна згрупувати (Group) (рис. 6.8), вирівняти (Align), налаштувати порядок (Order), рівномірно розподілити (Distribute) і закріпити (Lock) їх. Для роботи з декількома елементами можна також використати кнопки на панелі керування розміщені зверху.

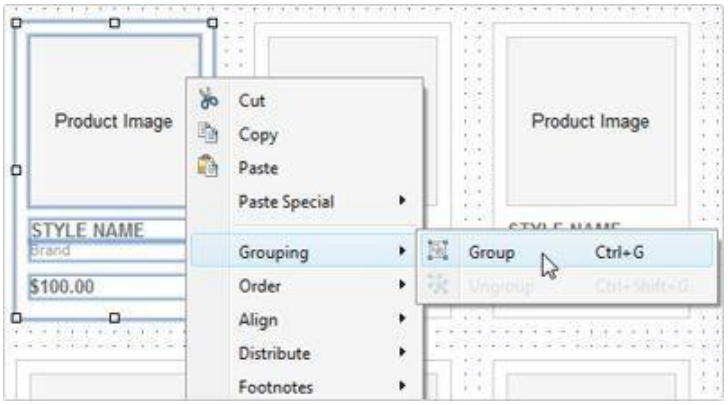


Рисунок 6.8 – Групування віджетів

Щоб відредагувати один з віджетів, не зачіпаючи при цьому інші елементи групи, в яку він входить, виберіть усю групу і натисніть на бажаний елемент.

Радіокнопки і групи вибору

Щоб додати радіокнопки в групу, виділіть їх, натисніть праву кнопку миші і виберіть опцію «Редагувати радіокнопку → Об'єднати в радіогрупу» (Edit Radio Button → Assign Radio Group). У групі радіокнопок одночасно може бути активна тільки одна радіокнопка.

Аналогічним чином, кнопки форм і зображень можна об'єднувати в групи вибору. Якщо активний один з елементів такої групи, то інші елементи групи автоматично набувають стандартних значень.

Редактор стилів віджетів

Редактор стилів віджетів (Widget Style Editor) дозволяє редагувати стандартне форматування віджетів і створювати власні стилі для централізованого керування відметами (рис. 6.9) (аналогічно стилям в Word).

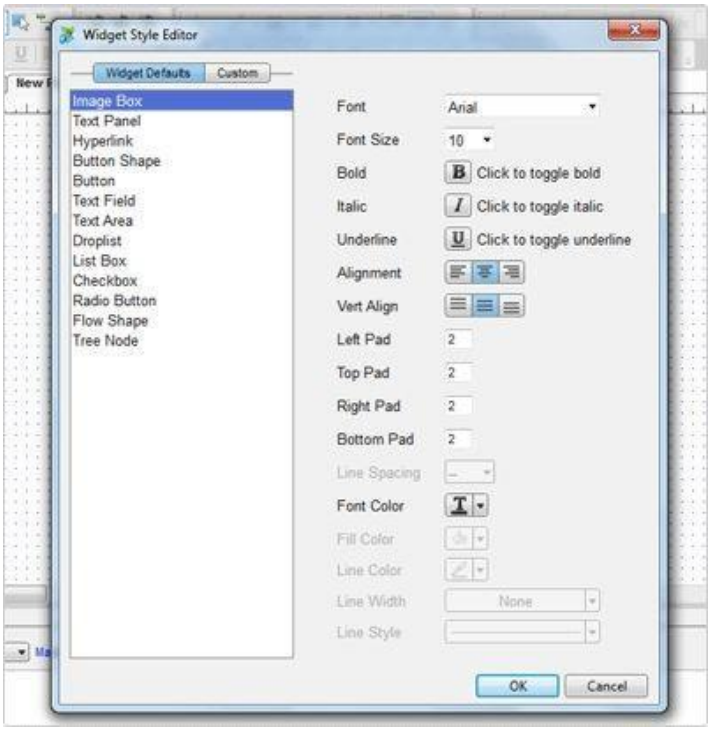


Рисунок 6.9 – Редактор стилів віджета

Щоб відкрити редактор стилів, натисніть кнопку «Widget Style Editor» на панелі інструментів (іконка з кнопкою «A» біля випадного списку, в якому вказаний стиль «Default»).

Редагування стандартного стилю віджета змінює усі елементи вибраного типу. Коли ви перетягуєте віджет з вікна бібліотеки віджетів у вікно створення макету, він створюється із стандартними параметрами форматування. Щоб швидко задати усім віджетам однакові параметри форматування, можна створити власні стилі. Щоб застосувати власний стиль до існуючого елементу, виберіть стиль з випадного списку в панелі редагування.

При редагуванні створеного раніше стилю, змінюється форматування усіх віджетів, до яких застосований цей стиль.

Форматування за зразком

Форматування за зразком (Format Painter) дозволяє копіювати параметри форматування одного віджета і застосовувати їх до інших (рис. 6.10). Воно працює як буфер обміну для параметрів форматування.

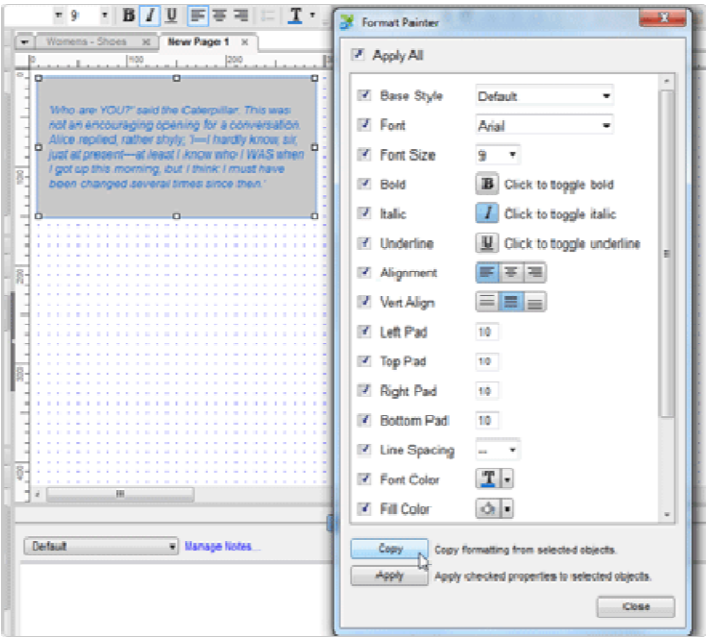


Рисунок 6.10 – Форматування за зразком

Щоб скопіювати параметри елементу, виберіть потрібний елемент у вікні створення макету і натисніть на кнопку «Format Painter» на панелі інструментів (іконка з кистю). Відкриється діалогове вікно перенесення параметрів, де показані поточні властивості елементу. Якщо при відкритому вікні діалогу вибрати віджет і натиснути «Copy», то його параметри скопіюються у вікно «Форматування за зразком».

Можна вказати параметри, що мають бути скопійовані – для цього відмітьте потрібні параметри прапорцями. Щоб застосувати скопійовані параметри до інших елементів, виберіть їх у вікні створення макету і натисніть кнопку «Apply» в діалоговому вікні.

Зміна режиму виділення

У Axure RP є два режими виділення – виділення пересічених об'єктів (Select Intersected Mode) і виділення обведених об'єктів (Select Contained Mode) (рис. 6.11). Кнопки цих функцій розташовані на панелі інструментів.

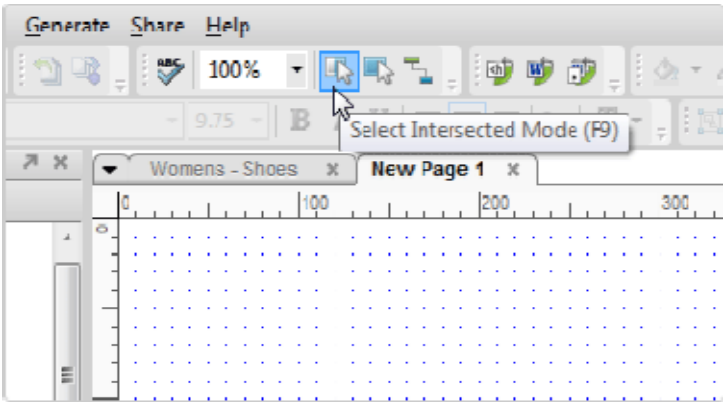


Рисунок 6.11 – Зміна режиму виділення

За замовчуванням завжди встановлений режим виділення пересічених об'єктів. У цьому режимі виділяються усі віджети, які хоча б частково потрапили в поле виділення.

6.4 Примітки

Примітки (Annotations) дозволяють додавати замітки до віджетів, з метою пояснення функціональності сторінки. На віджетах з примітками або взаємодіями у верхньому правому кутку з'являються спеціальні жовті ярлики-виноски з цифрами. Щоб приховати ці ярлики, досить зняти галочку з пункту Show Footnotes (Показувати виноски) в налаштуваннях макету в головному меню.

Додавання приміток

Щоб додати примітку, виберіть потрібний елемент у вікні створення макету і впишіть замітки в поля у вкладці «Annotations» у вікні властивостей віджета (Widget Properties Pane) (рис. 6.12).

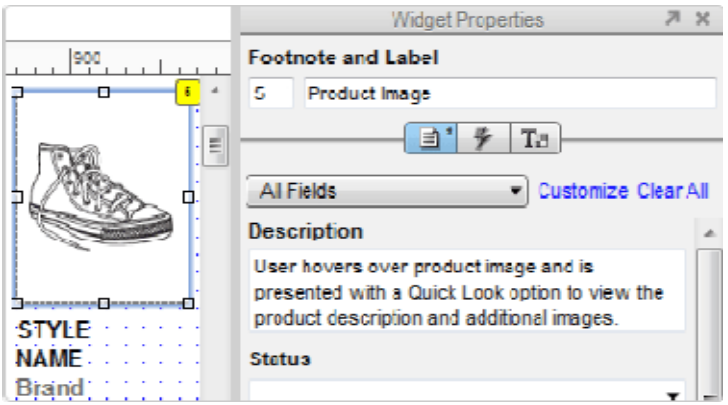


Рисунок 6.12 – Додавання нової примітки

Також можна виконати копіювання примітки одного віджета з метою використати їх в іншому віджеті. Для цього виділіть віджет і натисніть Ctrl+ C (або Cmd+C). Потім виділіть елементи, в які ви хочете вставити скопійовану примітку, натисніть на них правою кнопкою миші і виберіть в контекстному меню Paste Special → Paste Annotation (Вставити

примітку).Налаштування полів приміток

У головному меню виберіть Wireframe → Customize Annotations Fields and Views (Налаштування полів приміток і їх відображення) або натисніть на посилання Customize (Налаштувати) у вкладці Annotations у вікні Widget Properties. У відкритому діалоговому вікні налаштування полів приміток можна додавати, видаляти та змінювати порядок полів (рис. 6.13). Також можна створювати шаблони у вигляді різних наборів приміток для різних цілей.

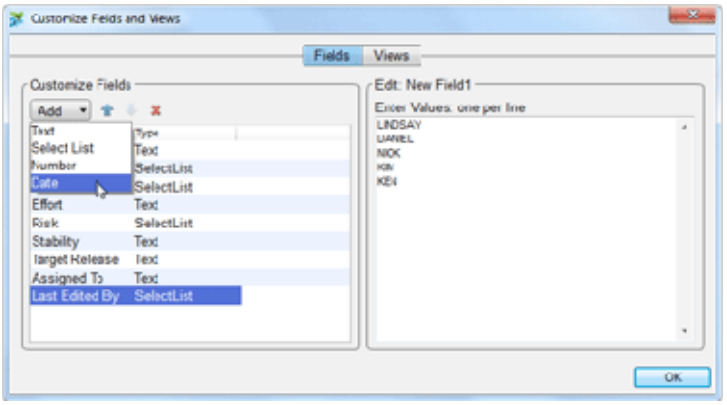


Рисунок 6.13 – Налаштування полів приміток

Зміна нумерації виносок

Щоб змінити номер виноски віджета, натисніть на ньому правою кнопкою миші і виберіть Footnotes → Decrement/Increment Footnote (Зменшити/Збільшити).

Також можна ввести потрібний вам номер виноски в поле порядкового номера виноски біля рядка назви у вікні Widget Properties. Проте, при цьому не забувайте, що елементи маркуються автоматично по порядку, так що номер, що вводиться, має бути меншим загальної кількості віджетів з виносками.

Додавання замiток до сторінок

Замітки до сторінок дозволяють збирати інформацію, яка відноситься до дизайну вашої сторінки. Замітки можна розбивати на декілька окремих полів (рис. 6.14). Це особливо зручно, якщо вам треба зробити окремі замітки для різних груп користувачів: для клієнтів, розробників або тестерів.

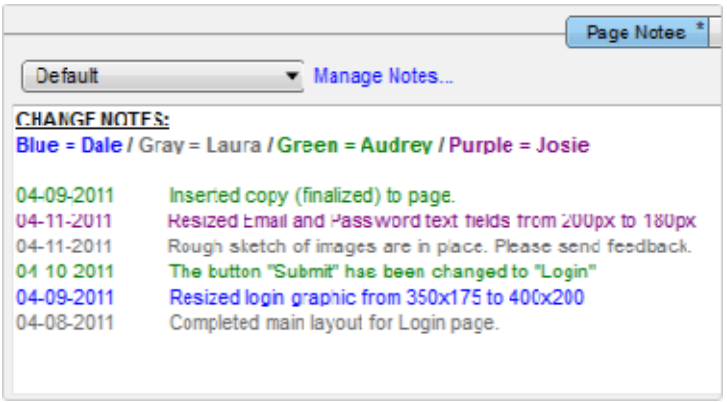


Рисунок 6.14 – Додавання замiток до сторінок

Вносити замітки можна в полі розміщеному під вікном створення макету сторінки.

Замітки можна форматувати за допомогою панелі редагування або стандартних комбінацій клавіш (наприклад, Ctrl/Cmd+B, Ctrl/Cmd+I, Ctrl/Cmd+U та іншими).

Додавання полів для замiток

У головному меню виберіть Wireframe → Manage Page Notes (Керування замітками) або

натисніть на посилання Manage Notes (Налаштувати замітки) на панелі налаштувань заміток до сторінки. За допомогою діалогового вікна «Замітки до сторінки» (Page Notes) ви можете створювати, перейменовувати і міняти місцями поля для заміток (рис. 6.15).

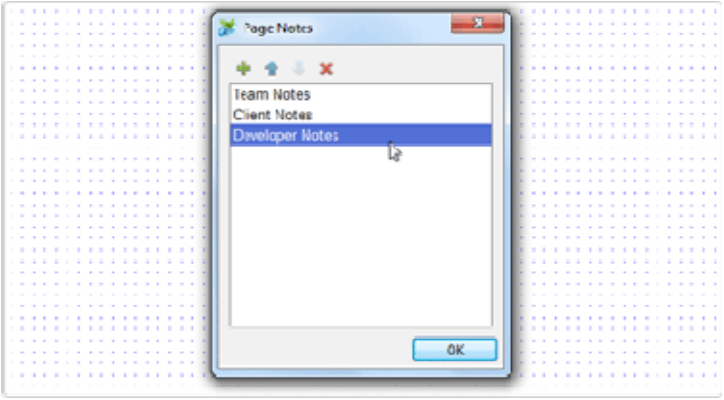


Рисунок 6.15 – Замітки до сторінки

Щоб переключитися між полями заміток, виберіть текстове поле замітки з випадаючого списку.

6.5 Форматування сторінки

Форматування сторінки (Page Formatting) дозволяє змінювати форматування окремих сторінок, а також застосовувати до сторінок стандартні та власні стилі. Опції форматування сторінки знаходяться у третій вкладці панелі властивостей сторінки (рис. 6.16).

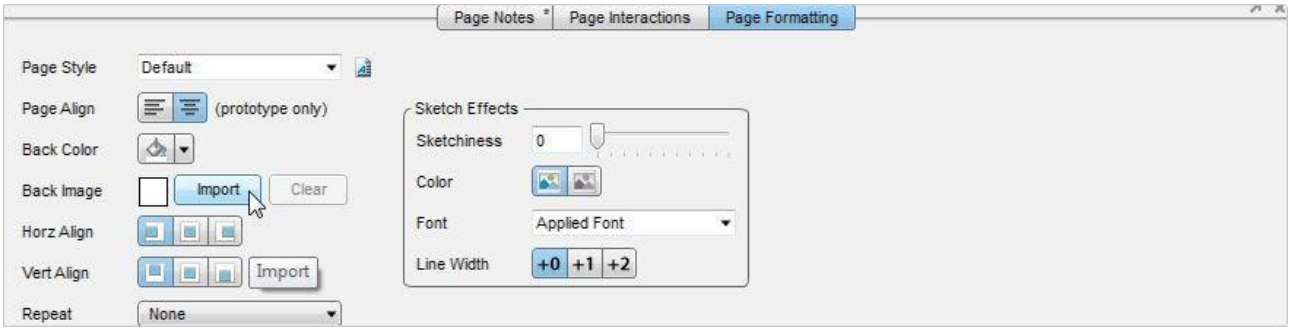


Рисунок 6.16 – Опції форматування сторінки

Параметри форматування сторінок

Форматування сторінки включає наступні опції:

1. **Стиль сторінки** (Page Style). Ви можете змінювати стилі сторінок так само, як і стилі віджетів. Щоб змінити стандартний стиль або створити власний, натисніть на кнопку праворуч від випадаючого списку, або виберіть Wireframe → Page Style Editor (Редактор стилів сторінки) в меню. Якщо вам треба розташувати кожну сторінку прототипу по центру екрану і додати стандартний фон, відредагуйте відповідним чином стиль, що використовується по замовчуванню, і він змінить усі сторінки.
2. **Вирівнювання сторінки** (Page Align). Сторінку прототипу можна вирівняти по лівому краю або по центру. Вирівнювання застосовується тільки до HTML, тому подивитися зміни у вікні створення макету в Axure RP неможливо.
3. **Колір фону** (Back Colour). Дозволяє задати фоновий колір сторінки.
4. **Фонове зображення** (Back Image). Дозволяє завантажити зображення і встановити його як фон.
5. **Горизонтальне і вертикальне вирівнювання** (Horiz Align and Vert Align). Дозволяє вирівняти фонове зображення по горизонталі та вертикалі.
6. **Повтор зображення** (Repeat). Дозволяє налаштувати повторення зображення горизонтально, вертикально, або в обох напрямах.

6.6 Ескіз

Ескіз (Sketch Effects) дозволяє надати будь-якому прототипу на будь-якому етапі його

створення вид нарису, зробленого від руки (рис. 6.17). Це може допомогти команді при необхідності зосередитися на архітектурі, інтерактивних компонентах та функціональності.

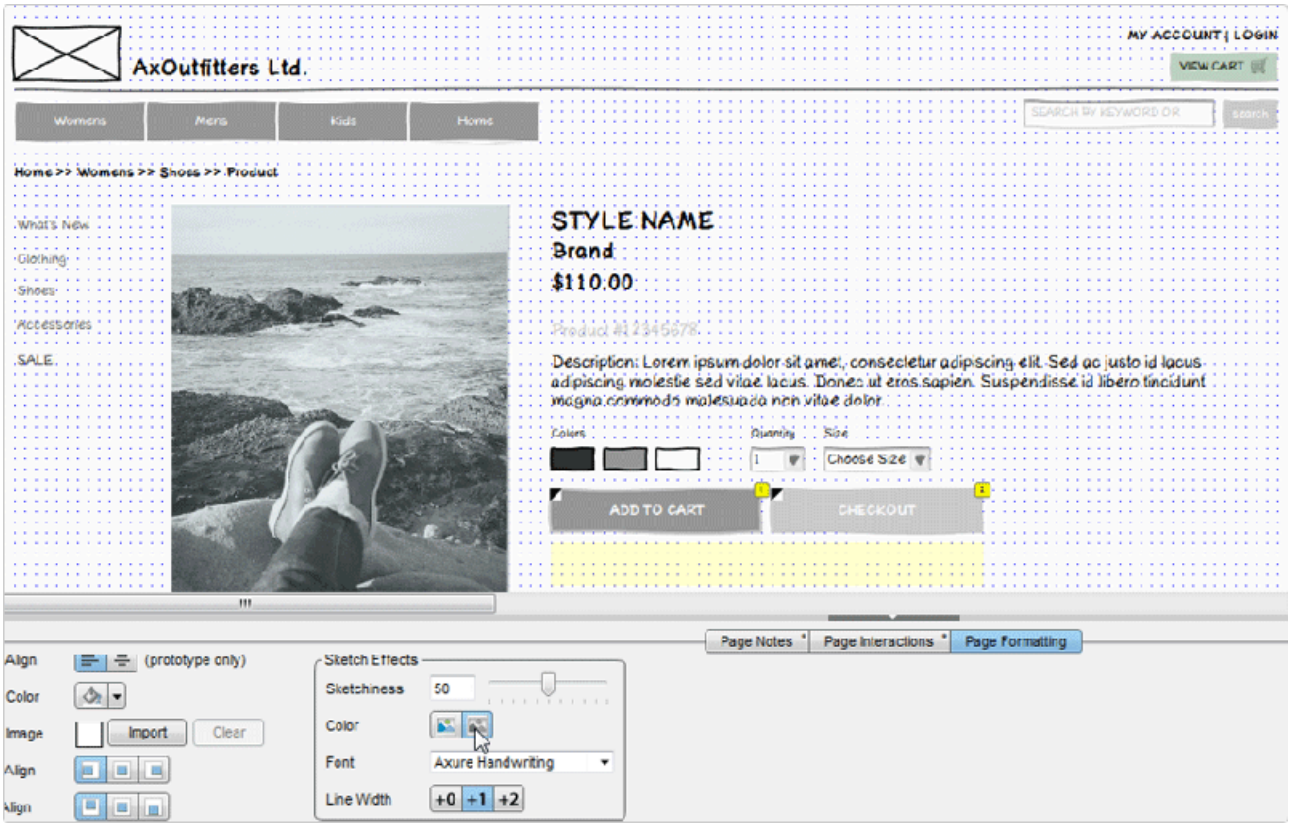


Рисунок 6.17 – Зразок ескізу сторінки

Ефекти ескіза входять до складу форматування сторінки (тому їх можна зберігати в стилях сторінки) і можуть бути налаштовані за допомогою наступних опцій:

1. **Схематичність** (Sketchiness). Чим вищий цей показник, тим більш хвилястими стають елементи інтерфейсу. Найчастіше встановлюється на рівні 50.
2. **Колір** (Color). Можна усю сторінку (у тому числі зображення, заливку, фон і текст) зробити чорно-білою.
3. **Шрифт** (Font). За допомогою цієї опції можна застосувати до усіх елементів на сторінці рукописні шрифти (наприклад, шрифт Axure Handwriting, Scoder Hand, Lucida Handwriting або Bradley Hand ITC).
4. **Ширина ліній** (Line Width). Для більшої схожості із справжнім ескізом, можна збільшити товщину ліній. Найчастіше використовується значення +1.

6.7 Загальні направляючі на сторінці

Направляючі лінії (Guides) допомагають підтримувати послідовність у сітці, а також розміщати віджети. Ви можете створити направляючі лінії як для окремих сторінок, так і для усіх сторінок в поточному файлі.

Додавання направляючих

Щоб додати направляючу лінію для поточної сторінки, досить перетягнути лінію від вертикальної або горизонтальної лінійки на полі створення макету (рис. 6.18). Зелений колір лінії означає, що направляюча зараз виділена. Зніміть виділення з лінії, і вона поміняє свій колір на синій.

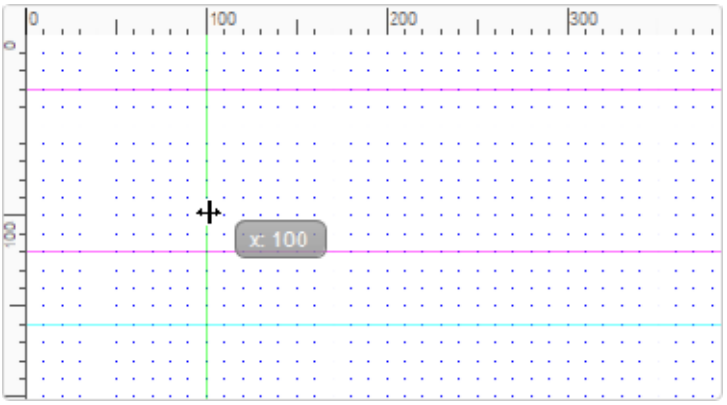


Рисунок 6.18 – Додавання направляючих ліній

Щоб додати загальну направляючу для усіх сторінок прототипу, затисніть Ctrl (або Cmd) і перетягніть лінію з лінійки. Зніміть виділення з лінії, і ви побачите, як вона поміняє свій колір на рожевий, що означатиме, що це загальна направляюча.

Щоб приховати або закріпити направляючу, натисніть праву кнопку миші у вікні створення макету і виберіть в контекстному меню Grid and Guides → Hide Global/Page Guides (Приховати загальні направляючі поточні направляючі) або Lock Guides (закріпити направляючі).

Якщо в рядку меню вибрати Wireframe → Grid and Guides → Create Guides (Створити направляючі), або вибрати цю опцію в контекстному меню, то відкриється діалогове вікно створення направляючих.

У ньому буде запропоновано вибрати одну з двох стандартних сіток: на 960 пікселів і 12 колонок або на 960 пікселів і 16 колонок. Також можна задати свої власні параметри. Якщо необхідно створити загальну направляючу для усіх сторінок прототипу, то виберіть опцію «Створити загальну направляючу».

6.8 Базова динаміка

У Axure RP можна створювати величезну кількість різних взаємодій – від звичайних гіпертекстових посилань до логічних операцій і математичних функцій.

Кількість необхідних в прототипі взаємодій визначається аудиторією і цілями прототипу. Іноді цілком достатньо і простого обговорення, проте, якщо ви збираєтеся проводити тестування, то вам доведеться створити прототип, що якнайбільш повно відображає фінальний результат. Перш ніж створювати складні взаємодії, необхідно переконатися в їх необхідності.

Взаємодії в Axure RP складаються з трьох частин: подій, сценаріїв і дій.

Події

Події (Events) – це тригери взаємодій. Клацання мишкою, наведення курсору, перетягання об'єкту – усе це події. Різні типи віджетів можуть генерувати різні типи подій:

1. **OnClick** – натиснення мишкою на віджет (працює з усіма віджетами, за винятком динамічної панелі).
2. **OnMouseEnter** – наведення курсору миші на віджет (зображення, текстовий блок, посилання, кнопка або активна область зображення).
3. **OnMouseOut** – переведення курсору миші з віджету (зображення, текстовий блок, посилання, кнопка або активна область зображення).
4. **OnKeyUp** – під час відпускання клавіші на клавіатурі (текстове поле і текстовий блок).
5. **OnFocus** – віджет отримує фокус під час клацання миші або кнопки табуляції (текстове поле, текстовий блок, випадаючий список, список, чекбокс та радіокнопка).
6. **OnLostFocus** – віджет втрачає фокус (текстове поле, текстовий блок, випадаючий список, список, чекбокс та радіокнопка.)
7. **OnChange** – у випадаючому списку або списку вибрано будь-який пункт.

Для динамічних панелей існують окремі події: OnMove, OnShow, OnHide,

OnPanelStateChange, OnDragStart, OnDrag і OnDragDrop.

Сценарії

Сценарії (Cases) – це можливі способи здійснення події. Наприклад, при натисненні на посилання може бути тільки один сценарій – відкриття іншої сторінки прототипу. При натисненні на кнопку входу в систему сценаріїв може бути два. Якщо вхід успішний, відкриється наступна сторінка, якщо ні – з'явиться повідомлення про помилку. У прототипі, створеному в Axure RP, сценарії можуть або бути представлені у вигляді опцій з описами, що пропонуються користувачеві після виконання події, або можна задати логіку, по якій автоматично виконуватимуться потрібні сценарії пов’язані із змінними.

Дії

Дії (Actions) – це реакції на події, визначені усередині сценарію. На прикладі гіперпосилання це виглядає так: при клацанні на гіперпосилання сторінка відкривається в поточному вікні. Значить, заданою дією було «відкрити сторінку в поточному вікні». Список можливих дій наступний:

1. Дії з посиланнями:
   - **Open Link in Current Window** – відкриває іншу сторінку або зовнішній URL в поточному вікні.
   - **Open Link in New Window/Tab** – відкриває іншу сторінку або зовнішній URL в новому вікні або новій вкладці.
   - **Open Link in Popup Window** – відкриває іншу сторінку або зовнішній URL в спливаючому вікні. Для цього вікна можна задати розміри і властивості.
   - **Open Link in Parent Window** – використовується в спливаючому вікні для того, щоб змінити сторінку, завантажену у батьківському вікні, з якого воно відкрите.
   - **Close Current Window** – закриває поточне вікно.
   - **Open Link(s) in Frame(s)** – змінює сторінку, завантажену у вбудований фрейм.
   - **Open Links in Parent Frame** – відкриває сторінку у батьківському фреймі. Використовується при переході із сторінки, завантаженої у вбудований фрейм.
2. Дії з динамічними панелями:
   - **Set Panel state(s) to State** – налаштовує видимість однієї або декількох динамічних панелей.
   - **Show Panel(s)** – відображає (робить видимою) одну або декілька динамічних панелей.
   - **Hide Panel(s)** – приховує одну або декілька динамічних панелей.
   - **Toggle Visibility** – приховує або відображає динамічні панелі залежно від їх поточного статусу видимості.
   - **Move Panel(s)** – пересуває динамічну панель в задане місце або на задану відстань.
   - **Bring Panel(s) to Front** – переміщає динамічну панель на самий верхній шар сторінки.
   - **Send Panel(s) to Back** – переміщає динамічну панель на самий нижній шар сторінки.
3. Дії з віджетами і змінними:
   - **Set Variable/Widget value(s)** – встановлює значення однієї або декількох змінних і/або віджетів (тобто значення тексту у віджеті).
   - **Scroll to Image Map Region** – прокручує сторінку до активної області зображення. Схоже на використання якоря або посилання переходу.
   - **Enable Widget(s)** – робить активними віджети форми.
   - **Disable Widget(s) –** робить неактивними віджети форми.
   - **Set Widget(s) to Selected State** – налаштовує вибраний стиль віджета або повертає його стандартний стиль.
   - **Set Focus on Widget** – перекладає фокус на віджет форми (наприклад, в текстове поле).
   - **Expant Tree Node(s)** – розгортає вузол в дереві віджетів.

- **Collapse Tree Node(s)** – згортає вузол в дереві віджетів.

4. Загальні дії:
   - **Wait Time (ms)** – відкладає дії на певний час.
   - **Other** – показує текстовий опис дії, наприклад «Відправити лист користувачеві».

6.9 Редактор сценаріїв

Додавання взаємодій

Щоб дізнатися можливі події для певного віджета, виберіть його та натисніть на вкладку «Interactions» у вікні властивостей віджета.

Щоб додати сценарій, натисніть «Add Case» (Додати сценарій) або двічі клацніть по події. Після цього відкриється редактор сценаріїв (Case Editor), в якому ви зможете вибрати і налаштувати потрібні дії (рис. 6.19).

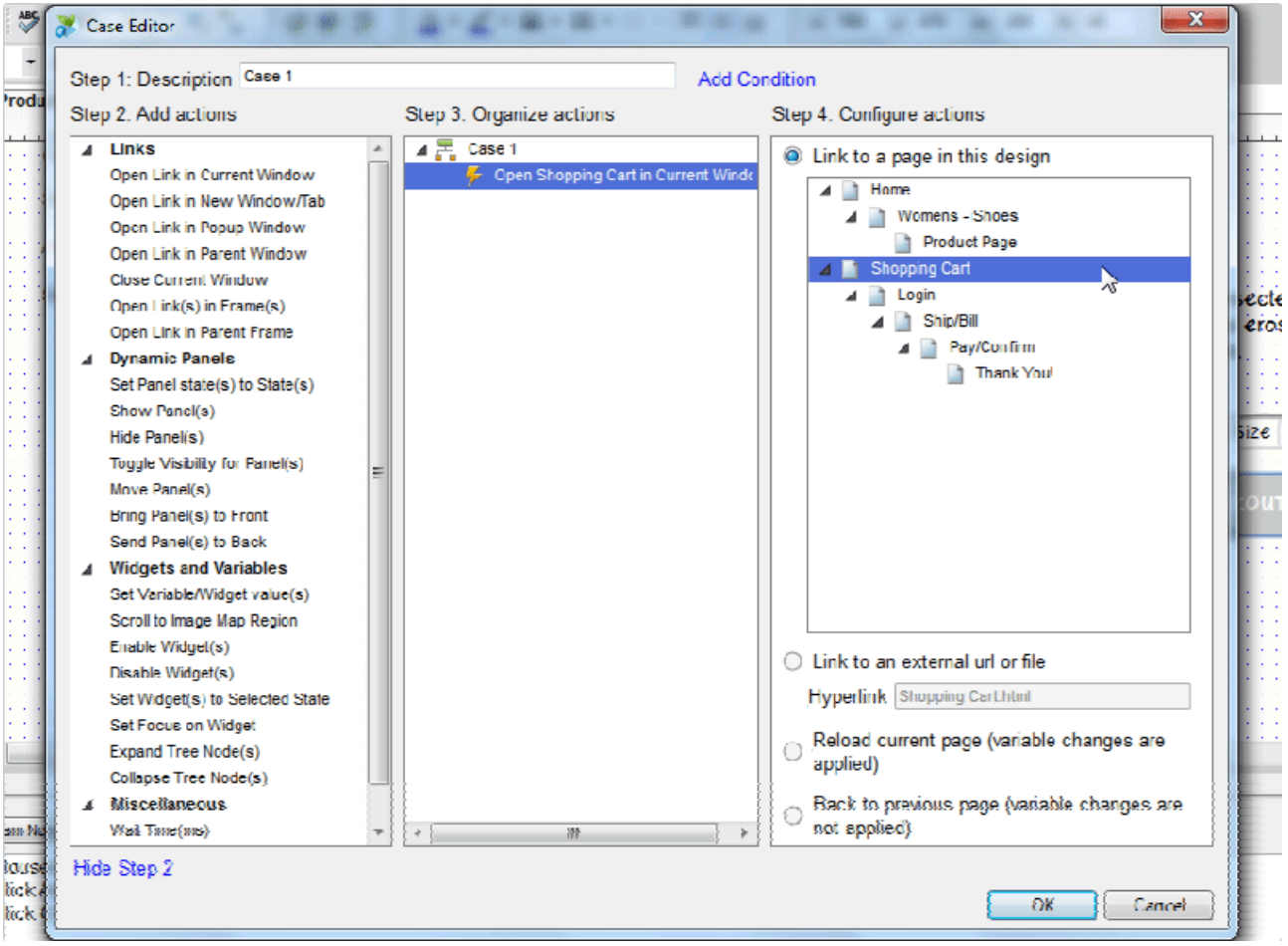


Рисунок 6.19 – Редактор сценаріїв

Вгорі (Крок 1: Опис (Description)) ви можете додати опис сценарію. Опис відображається в прототипі, якщо події присвоєно декілька сценаріїв, і не задана логіка, по якій вибирається виконуваний сценарій.

Для того, щоб додати в сценарій одну або декілька дій, виберіть потрібні дію із списку ліворуч (Крок 2: Додавання дій (Add actions)).

Вибрані дії відобразяться у середньому стовпці (Крок 3: Організація дій (Organize Actions)). Одну дію можна додавати декілька раз. Додані дії виконуватимуться в порядку розміщення у списку. Наприклад, якщо ви додасте дію «Встановити значення змінної» після дії «Відкрити посилання в поточному вікні», браузер відкриє посилання перед тим, як значення змінної зміниться.

Дії можна міняти місцями, перетягуючи їх або за допомогою контекстного меню, що викликається натисненням на стрілку біля дії або натисненням правої клавіші миші.

Для того, щоб налаштувати параметри дії в правій колонці, виберіть її в Кроці 3 (Крок 4: Конфігурація дій (Configure actions)).

Після того, як буде вибрано усі дії, натисніть кнопку «Ok» і додані сценарії та дії з'являться у вікні налаштування віджетів.

Налаштування множинних сценаріїв

Можливі випадки, коли необхідно буде зробити так, щоб одна подія в різних випадках викликала різні сценарії. Щоб привласнити події додаткові сценарії, необхідно повторити ті ж кроки, що і при додаванні сценарію. Наприклад, при виконанні події OnClick на кнопці, ви можете створити два сценарії з описами «Успішна авторизація» і «Невдала авторизація». Клік по кнопці в прототипі покаже описи, і користувач прототипу зможе вибрати одну з двох можливих дій (рис. 6.20).

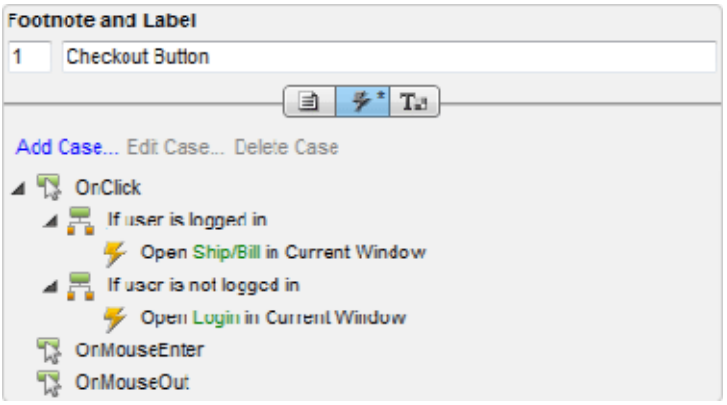


Рисунок 6.20 – Налаштування множинних сценаріїв

Грамотно прописані описи сценаріїв – це ефективний спосіб передати словами логіку роботи веб-сайту. Крім того, описи легко виправляти. Якщо вам треба, щоб в прототипі потрібний сценарій вибирався автоматично, ви можете задати логіку, згідно якої виконувані сценарії вибиратимуться на основі значень змінних або даних, введених користувачем в згенерованому прототипі.

Завдання на лабораторну роботу

1. Ознайомитися з теоретичними відомостями по роботі.
2. Виконати приклади з уроків розміщених на офіційному сайті:
   - ✔ My First Prototype (http://www.axure.com/learn/core/first-prototype);
   - ✔ Linking to Pages (http://www.axure.com/learn/basic/interactions/page-links-tutorial);
   - ✔ Multiple Cases (http://www.axure.com/learn/basic/interactions/multiple-cases-tutorial);
   - ✔ Anchor Links (http://www.axure.com/learn/basic/interactions/anchor-links-tutorial).
3. Зробити висновки по роботі.

КОНТРОЛЬНІ ЗАПИТАННЯ

1.Що таке прототип програмного продукту?
2.Які ви знаєте форми представлення прототипів програмного продукту?
3.Для чого розробляють прототип?
4.Якими засобами можна розробити прототип програмного продукту?
5.Що таке модель програмного забезпечення?
6. Які оції включає форматування сторінки?:

ЛАБОРАТОРНА РОБОТА № 7

Робота з динамічними об’єктами засобами Axure RP

Мета роботи: Навчитися використовувати динамічні панелі для приховування, відображення, переміщення та зміни віджетів прототипу. Отримати навички по роботі із майстрами у Axure RP. Навчитися створювати інтерактивні прототипи шляхом налаштування інструментів для їх генерації.

Теоретичні відомості

7.1 Робота з динамічними панелями

Віджет «Динамічна панель» – це набір режимів або діаграм, які включають інші віджети. Динамічну панель можна приховувати, відображати або переміщати. Поточний стан панелі може змінюватися динамічно. Усе це дозволяє демонструвати такі функції прототипу, як: призначені для користувача підказки, лайтбокси, вкладки, функції перетягування (Drag and Drop). Як правило, в Axure RP для приховування, відображення, переміщення або зміни віджетів прототипу використовуються динамічні панелі. Тому динамічна панель є одним із самих використовуваних елементів програми.

Стани динамічної панелі

У динамічної панелі може бути один або декілька станів, і кожен стан є діаграмою, що включає інші віджети. В один момент часу видимим може бути тільки один стан динамічної панелі. Використовуючи взаємодії, можна приховувати і відображати панель, а також перемикати її поточний стан.

Для того, щоб додати або змінити розмір динамічної панелі, необхідно перетягнути віджет Dynamic Panel з вікна бібліотеки віджетів у вікно створення макету. Розмір віджета динамічної панелі визначає видимі межі станів панелі на сторінці макету.

Якщо у вікні створення макету вже є віджети, які ви б хотіли зробити динамічними, то для цього можна використати функцію «Конвертувати в Динамічну панель». Виберіть віджети, які ви хотіли б помістити в динамічну панель, клікніть по них правою кнопкою миші і виберіть Convert → Convert to Dynamic Panel. Ця дія автоматично створить нову динамічну панель і помістить вибрані елементи у перший стан цієї панелі.

Додавання і оформлення станів динамічних панелей

За замовчуванням динамічні панелі порожні, тому їх стани необхідно наповнити віджетами. Для цього подвійним клацанням на панелі відкрийте «Менеджер станів динамічних панелей» (рис. 7.1).

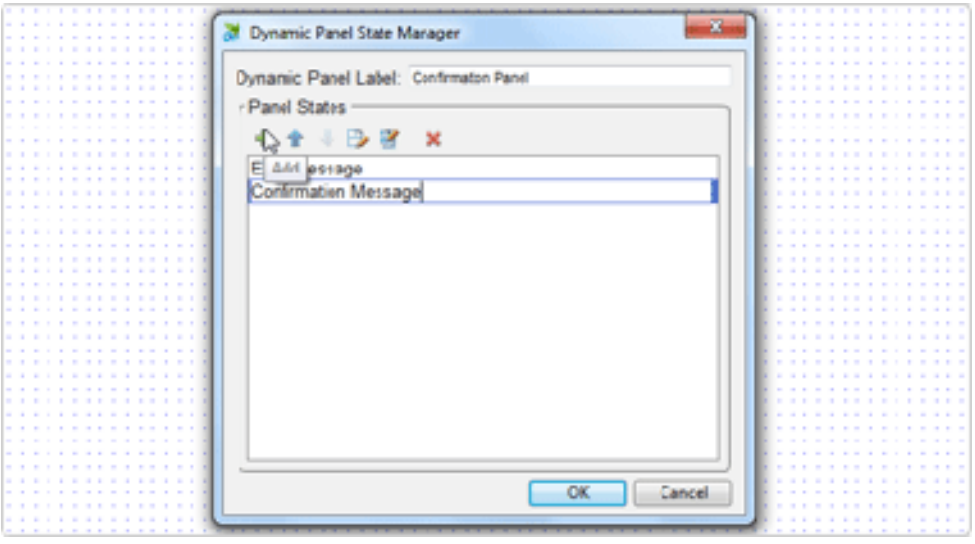


Рисунок 7.1 – Менеджер станів динамічних панелей

У цьому діалоговому вікні ви зможете додати, видалити, перейменувати і поміняти місцями стани, а також відкрити їх для редагування. Перший стан в списку є стандартним.

Щоб відредагувати один із станів, виберіть його в списку і натисніть Edit State (Редагувати стан) (рис. 7.2).

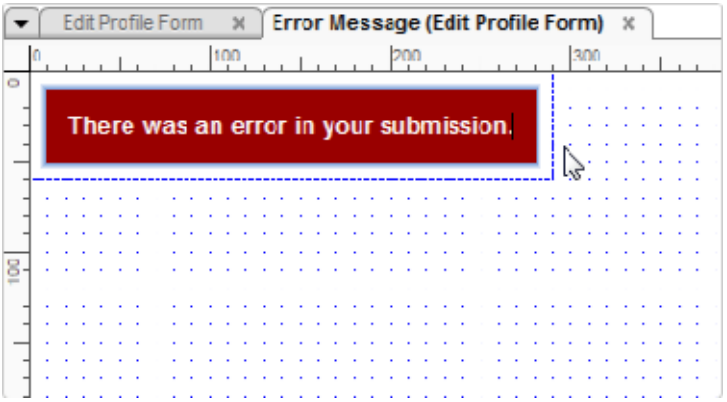


Рисунок 7.2 – Приклад редагування стану динамічної панелі

Редагований стан має синю пунктирну рамку. Ця рамка відображає розміри динамічної панелі і є межею області, яку буде видно на сторінці. Щоб наповнити стан панелі віджетами, перетягніть їх мишкою.

Щоб додати смугу прокрутки на динамічну панель, натисніть на панелі правою кнопкою миші і виберіть Edit Dynamic Panel → Show Scrollbars as Needed (Показувати смугу прокрутки за потреби). Якщо в цьому стані панелі ви помістите віджети, які виходять за межі видимої області динамічної панелі, то всередині динамічної панелі прототипу з'явиться смуга прокрутки.

Приховування панелі за замовчуванням

Динамічні панелі можна зробити прихованими за замовчуванням. Для цього натисніть на панелі правою кнопкою миші і виберіть в контекстному меню Edit Dynamic Panel → Set Hidden (Зробити прихованою) (рис. 7.3).

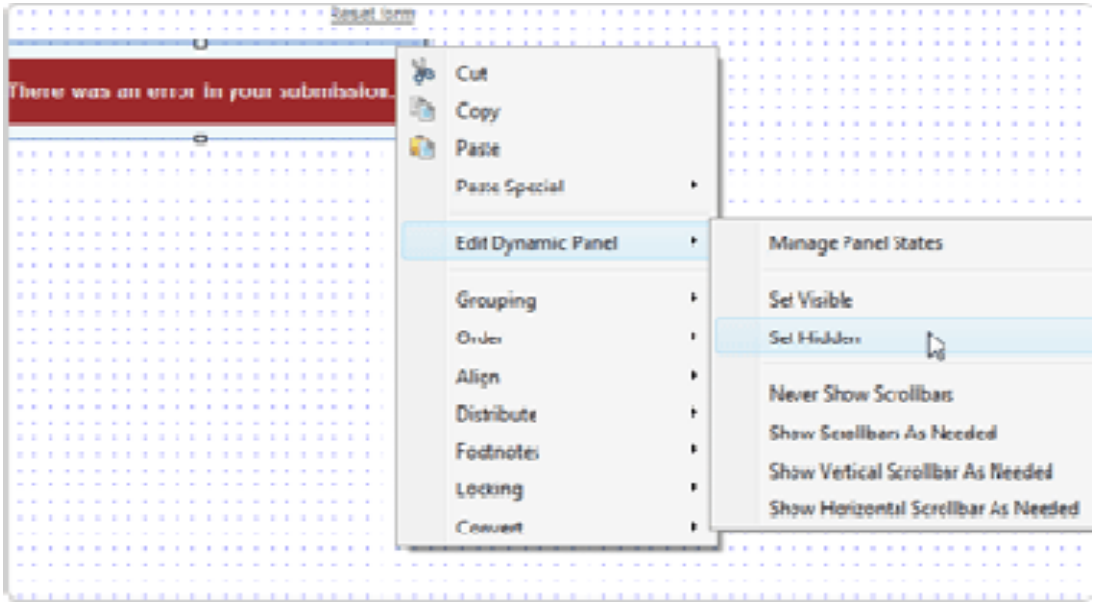


Рисунок 7.3 – Приховування динамічної панелі

Прихована панель міняє колір віджета з блакитного на жовтий.

Вікно керування динамічними панелями

Вікно керування динамічними панелями (Dynamic Panel Manager) показує усі динамічні панелі, наявні в поточному прототипі і надає швидкий доступ до керування ними. Ця функція корисна у випадках, коли на сторінці використовується багато динамічних панелей, між якими треба швидко переміщатися, а також за її допомогою можна приховувати

динамічні панелі з метою доступу до елементів під панеллю (рис. 7.4).

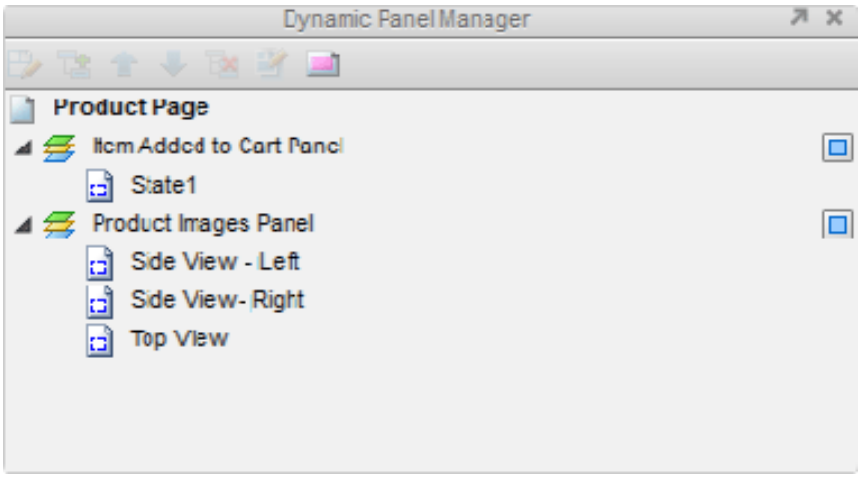


Рисунок 7.4 – Вікно керування динамічними панелями

Також вікно керування динамічними панелями містить опцію, що включає відображення в ньому майстрів, розташованих на поточній сторінці.

Якщо це вікно не відображається, ви можете включити його через меню (View → Dynamic Panel Manager).

Особливості використання вікна керування динамічними панелями

Щоб додати стан динамічної панелі, виберіть панель і натисніть кнопку Add State (Додати стан) в панелі інструментів цього вікна. Ви також можете натиснути правою кнопкою миші на назву панелі або її стану і використати контекстне меню.

Щоб відкрити стан для редагування, двічі кликніть по ньому. Також можна виділити панель або стан і натиснути Edit All States (Редагувати усі стани) – ця опція відкриє редагування усіх станів динамічної панелі. Також ця опція доступна через контекстне меню правої кнопки миші (рис. 7.5).

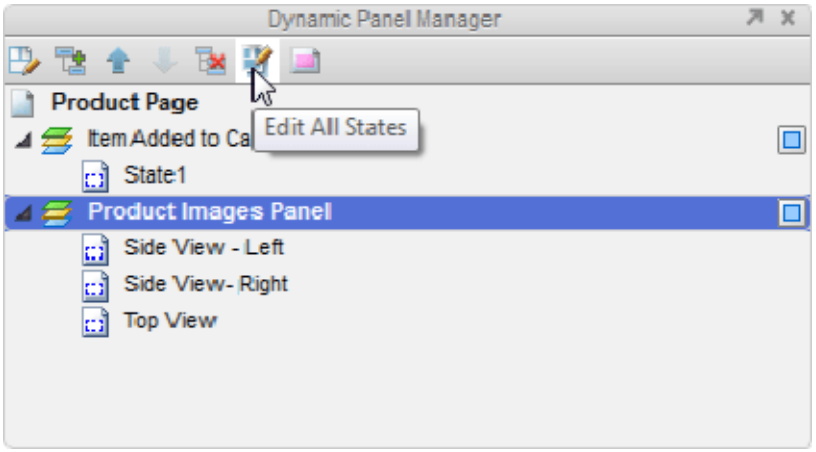


Рисунок 7.5 – Редагування стану динамічної панелі

Щоб приховати динамічну панель у вікні створення макету, натисніть на блакитний квадратик праворуч від назви панелі. Щоб приховати усі динамічні панелі на сторінці, натисніть правою кнопкою на імені панелі і виберіть опцію Hide All (Приховати усе). Ця функція корисна у разі, якщо динамічні панелі перекривають собою інші елементи.

Взаємодії з динамічними панелями

До динамічної панелі можна прив'язати взаємодію, додавши сценарій до події і вибравши відповідну дію із списку. Для динамічних панелей доступні наступні дії: Set Panel state(s) to State(s), Toggle Visibility for Panel(s), Show Panel(s), Move Panel(s), Hide Panel(s), Bring Panel(s) to Front, Send Panel(s) to Back.

Розглянемо декілька простих взаємодій, включаючи приховування/відображення динамічних панелей, перемикання станів динамічних панелей і переміщення динамічних

панелей на самий верхній шар сторінки.

Перемикання стану динамічної панелі

Створіть динамічну панель з декількома станами, після чого застосуєте до неї дію Set Panel state(s) to State(s) щоб встановити панель в потрібний стан. У Редакторі сценаріїв виберіть цю дію, а потім виберіть відповідну панель із списку (рис. 7.6). Після цього ви зможете вибрати стан, в який панель повинна буде перейти в цьому сценарії. У одній і тій же дії можна задати стани декількох динамічних панелей.

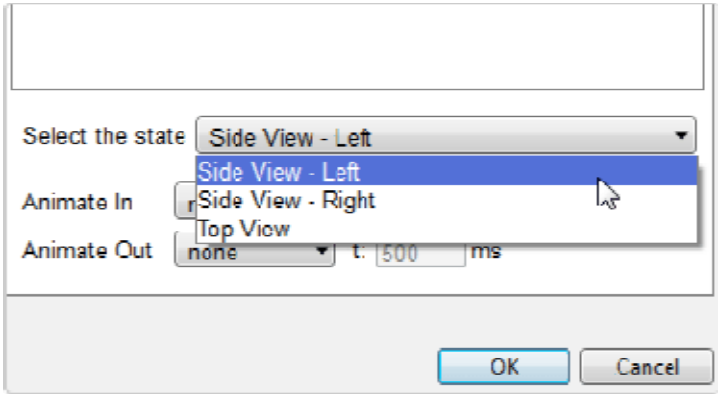


Рисунок 7.6 – Перемикання стану динамічної панелі

Керування видимістю панелі

Щоб приховати або відобразити зміст поточного стану динамічної панелі, використайте дії Show Panel(s) і Hide Panel(s). У редакторі сценаріїв виберіть потрібну дію та виберіть панель із списку на поточній сторінці (рис. 7.7). Однією дією можна приховувати або відображати на сторінці відразу декілька панелей.

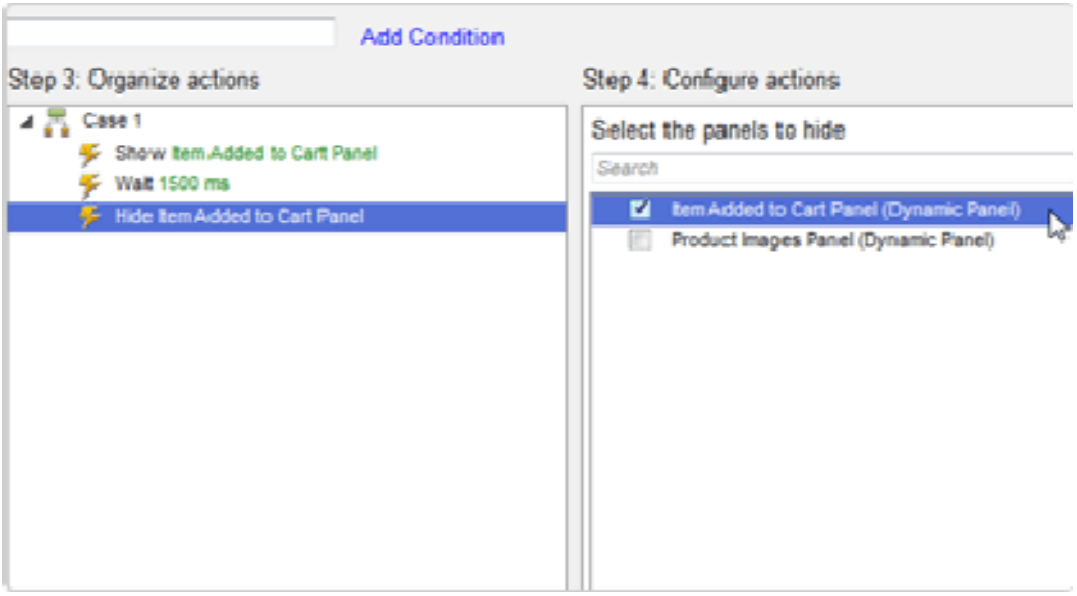


Рисунок 7.7 – Керування видимістю панелі

Для послідовного перемикання видимості динамічної панелі використовується дія Toggle Visibility for Panel(s). Ця дія може застосовуватися для таких елементів, як повідомлення про помилки, які відображаються та приховуються динамічно.

Переміщення динамічної панелі

Дія Move Panel(s) дозволяє динамічно переміщати панель на певну відстань задану у пікселях або переміщати її в певне положення на сторінці (рис. 7.8). Помістіть в динамічну панель елементи, які повинні рухатися, і задайте сценарій події, яка ініціює рух панелі (наприклад, це може бути подія OnClick для кнопки).

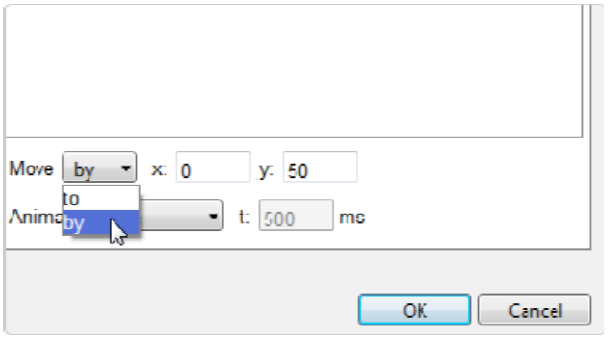


Рисунок 7.8 – Переміщення динамічної панелі

У Редакторові Сценаріїв виберіть дію Move Panel(s) і потім виберіть панель, яка повинна рухатися. Під колонкою з Кроком 4 можна встановити відстань у пікселях, на яку слід перемістити панель (по координатах X або Y) або задати конкретне положення (по координатах X та Y) в яке її необхідно перемістити.

Цю дію можна використати для створення дій розкривання і приховування блоків, які спочатку приховані за іншим контентом або просто невидимі.

Переміщення панелі на самий верхній або самий нижній шар

Якщо динамічна панель знаходиться за іншими елементами і її необхідно перемістити на верхній шар, то можна використати дію Bring Panel(s) to Front (рис. 7.9). У редакторі сценаріїв виберіть необхідну дію і потрібну панель. У одній дії можна вибрати відразу декілька панелей, які перемістяться на верхній шар сторінки. Дія Send Panel(s) to Back переміщає панель на самий нижній шар.

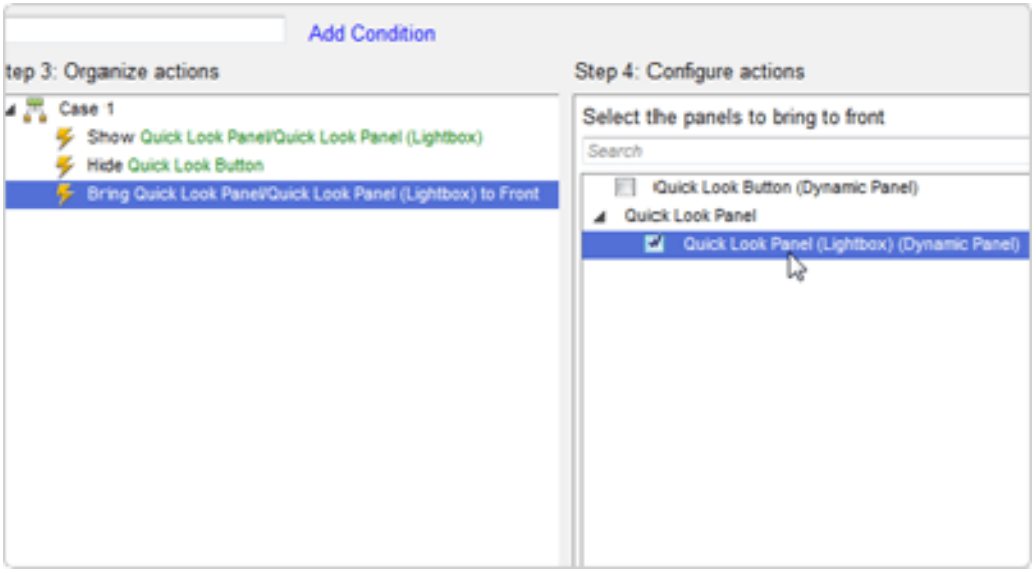


Рисунок 7.9 – Переміщення панелі на самий верхній або самий нижній шар

Подібні дії можуть використовуватися для створення лайтбоксів, випадаючих меню, тултипів, у випадках, коли панель знаходиться на нижньому шарі сторінки.

Анімація і ефекти переходу

За допомогою анімації можна налаштувати переміщення динамічних панелей, а також приховати або відобразити їх з ефектом плавного згасання і появи. Задаючи дію Move Panel, ви можете вибрати ефект анімації, як наприклад поворот або стрибок, і визначити тривалість анімації (рис. 7.10).

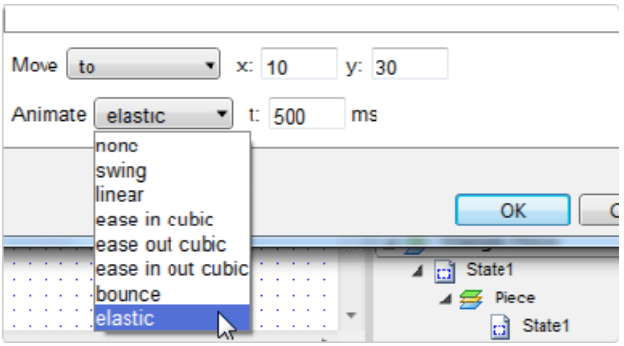


Рисунок 7.10 – Анімація і ефекти переходу

А для дій Show Panel, Hide Panel і Toggle Panel Visibility можна налаштувати ефект плавного згасання або появи.

Анімацію перемикання станів панелі можна налаштувати так, щоб стани «переходили» з одного в інший або щоб перехід здійснювався за допомогою плавного згасання першого стану і плавної появи іншого.

7.2 Використання майстрів

Майстри – це набори віджетів, які багаторазово можна використати в одному файлі. Найчастіше майстри створюють для роботи з хедерами, футерами, навігацією і шаблонами сторінок.

Головна перевага майстрів полягає в тому, що один майстер можна застосовувати одночасно до декількох сторінок, при цьому редагуючи його в одному місці. Будь-які зміни, які ви вносите при редагуванні майстра, автоматично застосовуються на усіх сторінках, які його використовують. На одну сторінку можна додавати необмежену кількість майстрів.

Панель майстрів

Створити і налаштувати майстер можна за допомогою панелі майстрів, розташованої в лівому нижньому куті.

Щоб створити майстер, кликніть по кнопці Add Master (Створити майстер) на панелі інструментів вікна майстрів (рис. 7.11).

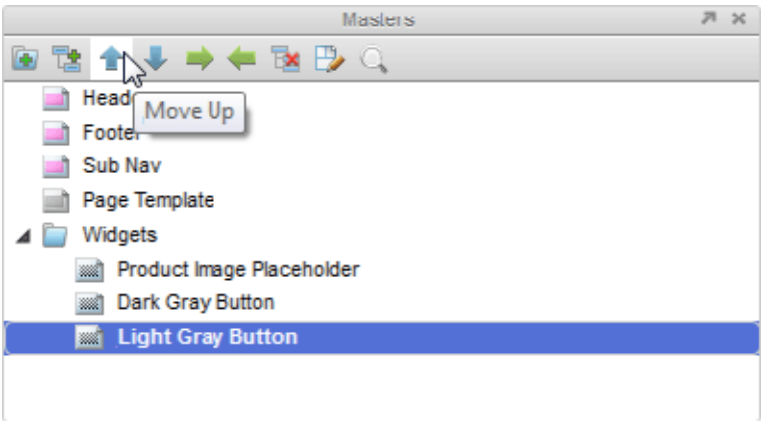


Рисунок 7.11 – Панель майстрів

Щоб перейменувати майстер, клікніть по ньому двічі з паузою. Щоб видалити майстер, виділіть його і натисніть кнопку Delete Master, розташовану на панелі інструментів вікна майстрів.

Для впорядковування майстрів їх можна перетягувати, або використовувати стрілки на панелі інструментів вікна майстрів. Також можна розмістити майстри по каталогах. Для цього створіть каталог за допомогою кнопки «Add folder» на панелі інструментів вікна майстрів, а потім перетягніть в нього потрібні майстри.

Редагування майстрів

Щоб відкрити майстер для редагування в області створення макету, відкрийте його подвійним клацанням (рис. 7.12).

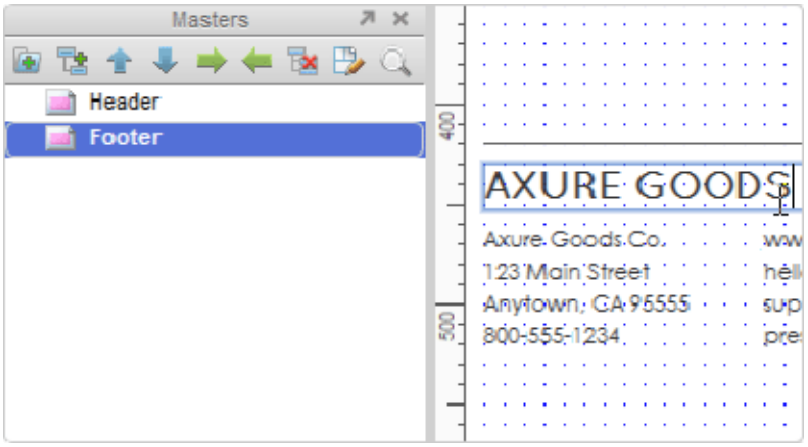


Рисунок 7.12 – Редагування майстрів

Додавання віджетів в майстер виконується шляхом перетягування на нього елементів. Також можна поміщати одні майстри в інші, перетягуючи існуючий майстер з вікна майстрів у вікно створення макету.

Якщо ви працюєте із сторінкою, і вирішили помістити декілька віджетів у майстер, виділіть ці віджети і в контекстному меню виберіть опцію Convert →Convert to Master (Конвертувати в майстер). Ця дія створить майстер, помістить в нього виділені елементи і розмістить майстер на поточній сторінці.

Поведінка майстрів

Майстри можуть наслідувати три поведінки: Нормальна (Normal), Користувацький віджет (Custom Widget) або Частина фону (Place in background). Щоб переключити поведінку майстра, треба натиснути правою кнопкою на назві майстра у вікні майстрів і вибрати поведінку через контекстне меню.

Ви можете змінювати поведінку будь-якого майстра у будь-який час, проте така зміна буде стосуватися майстера тільки у тому випадку, якщо він був доданий на сторінку після зміни. Наприклад, якщо у вас був майстер з поведінкою Custom Widget, і ви поміняли її на Normal, ця дія не поміняє поведінку цього майстра в місцях, у яких він уже був доданий.

Щоб змінити поведінку майстра, який вже знаходиться на одній із сторінок, клікніть по ньому правою кнопкою миші у вікні створення макету і перейдіть в підменю Master через контекстне меню. Там ви знайдете опції Place in Background (Помістити на фон), Remove from Background (Прибрати з фону) і Flatten (Вирівняти).

Нормальна (Normal) поведінка майстра (поведінка за замовчуванням)

Майстер з нормальною поведінкою можна переміщати по сторінці без обмежень (рис. 7.13).

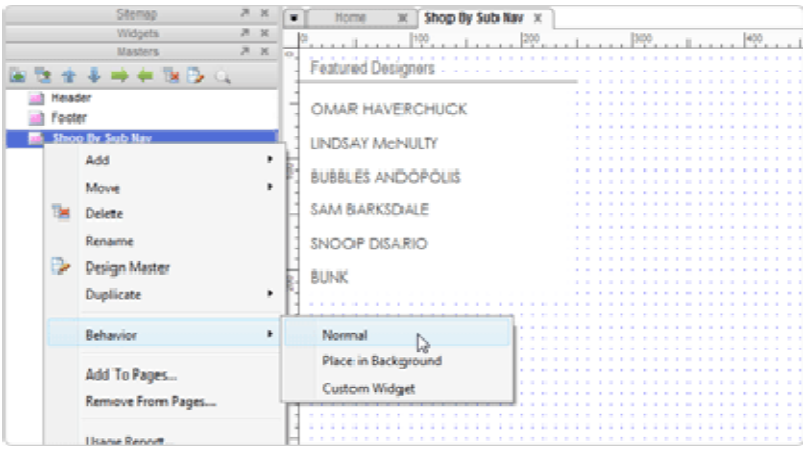


Рисунок 7.13 – Нормальна (Normal) поведінка майстра

Майстри з нормальною поведінкою мають рожеву маску, проте її можна вимкнути в рядку меню (виберіть Wireframe → Mask Masters).

Поведінка «Частина фону» (Place in Background)

Після додавання майстра з такою поведінкою на сторінку, він зливається з фоном сторінки (рис. 7.14). Такий тип майстрів зручний для створення шаблонів сторінок і блокової розмітки.

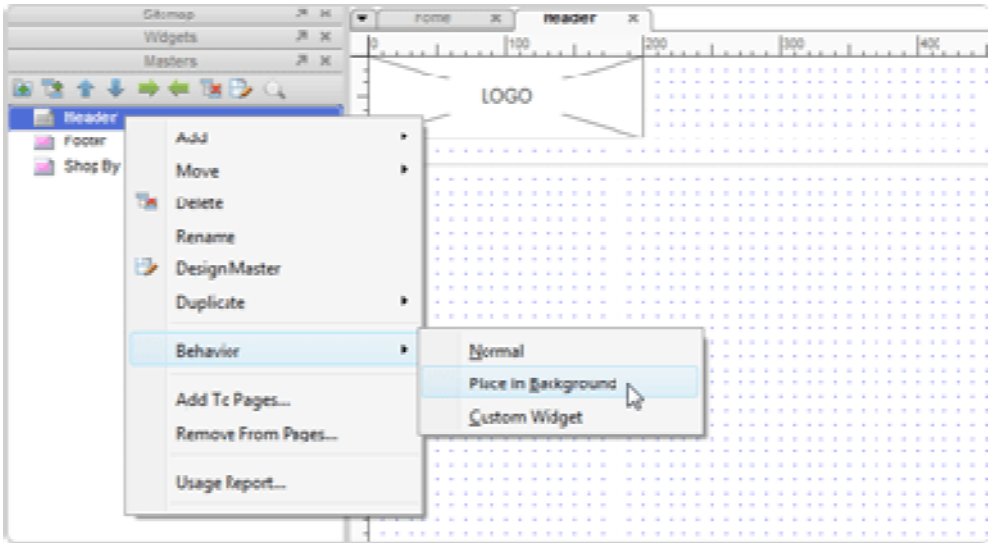


Рисунок 7.14 – Поведінка майстра «Частина фону»

Майстри з такою поведінкою мають сіру маску.

Поведінка «Користувацький віджет» (Custom Widget)

Майстер з таким типом поведінки втрачає зв'язок із первинним майстром і може бути відредагований окремо як будь-який інший відмет (рис. 7.15). Це зручно при створенні бібліотеки віджетів із заздалегідь заданими властивостями, примітками і/або взаємодіями.

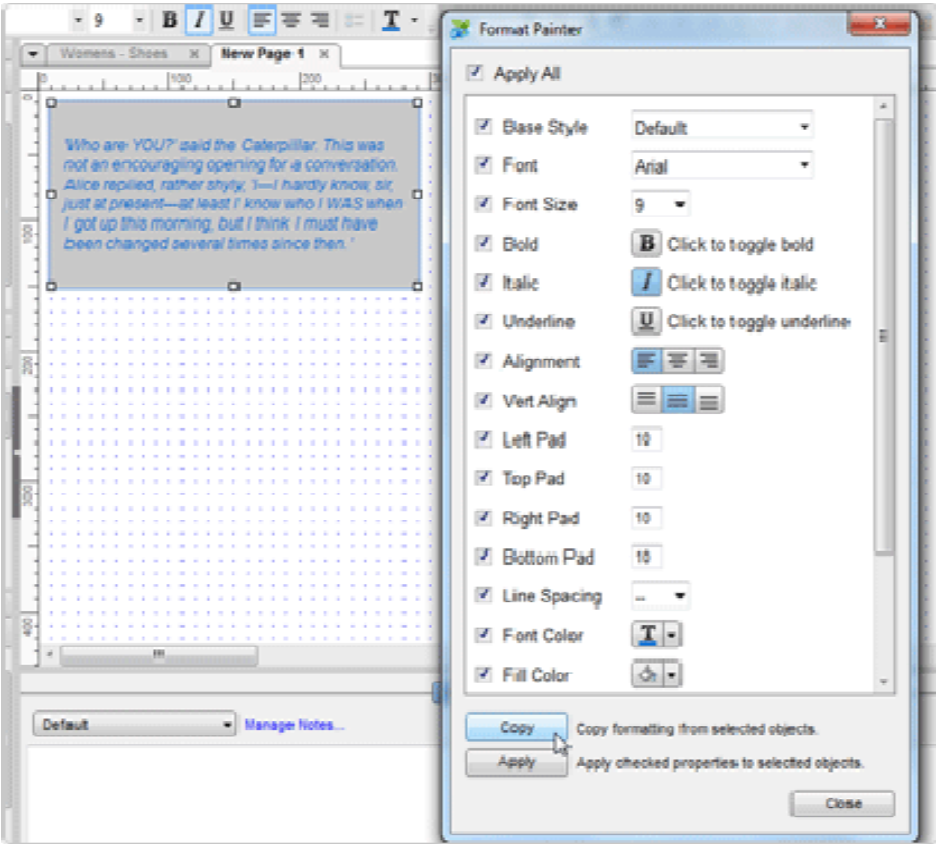


Рисунок 7.15 – Поведінка майстра «Користувацький віджет»

Майстри зберігаються у файлі проекту. Щоб створювати бібліотеки, які зберігаються окремо, і які можна використати в інших проектах, необхідно створити призначену для користувача бібліотеку елементів.

Додавання майстрів на сторінки

Щоб додати майстер на сторінку, перетягніть його з вікна майстрів у вікно створення макету (рис. 7.16).

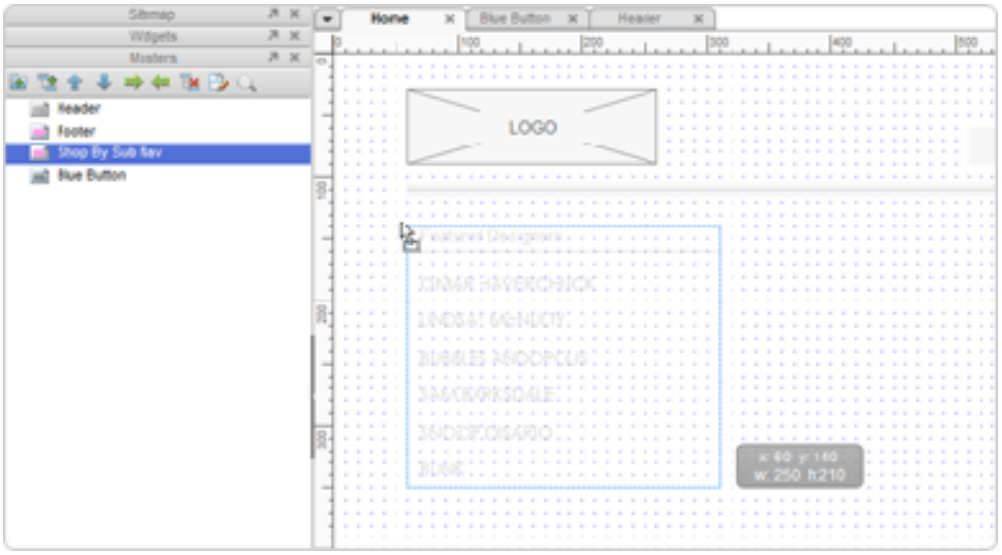


Рисунок 7.16 – Додавання майстрів на сторінки

У кожного майстра є своя колірна маска: майстер з нормальною поведінкою відображається рожевим, а майстер з поведінкою «Частина фону» – сірим. Виділення маскою можна відключити у рядку меню, знявши галочку з опції Mask Masters в меню Wireframe.

Додавши майстер у вікно створення макету, ви можете відкрити його для редагування подвійним клацанням у вікні майстрів або у вікні створення макету.

7.3 Створення інтерактивних прототипів

Інтерактивні прототипи – це прекрасний спосіб спростити документацію, отримати зворотний зв'язок з користувачами, визначити і скласти необхідні технічні вимоги. Після створення макету з примітками і взаємодіями, ви можете створити інтерактивний прототип, що буде працювати у браузері, без написання додаткового коду.

Інтерактивні прототипи в Axure RP створюються з використанням HTML, JavaScript, CSS та файлів зображень, а тому їх можна переглядати в усіх популярних браузерах: Firefox, Internet Explorer, Opera і Chrome. Для перегляду згенерованого прототипу у браузері немає необхідності встановлювати спеціальний рідер, плеєр або Axure RP, що значно спрощує його демонстрацію користувачам.

Згенерований код, як правило, не дуже добре підходить для випуску в якості кінцевого продукту, проте відкритий API дозволяє працювати з генератором коду.

Налаштування прототипу

Програма містить велику кількість опцій генерації прототипу, серед яких є вибір сторінок, а також створення логотипу або супровідного підпису. Після налаштування прототипу, генерація HTML і JavaScript коду здійснюється натисненням кнопки Generate.

Щоб викликати діалогове вікно генерації прототипу, виберіть в рядку меню Generate → Prototype або натисніть на кнопку Prototype в панелі інструментів (рис. 7.17).

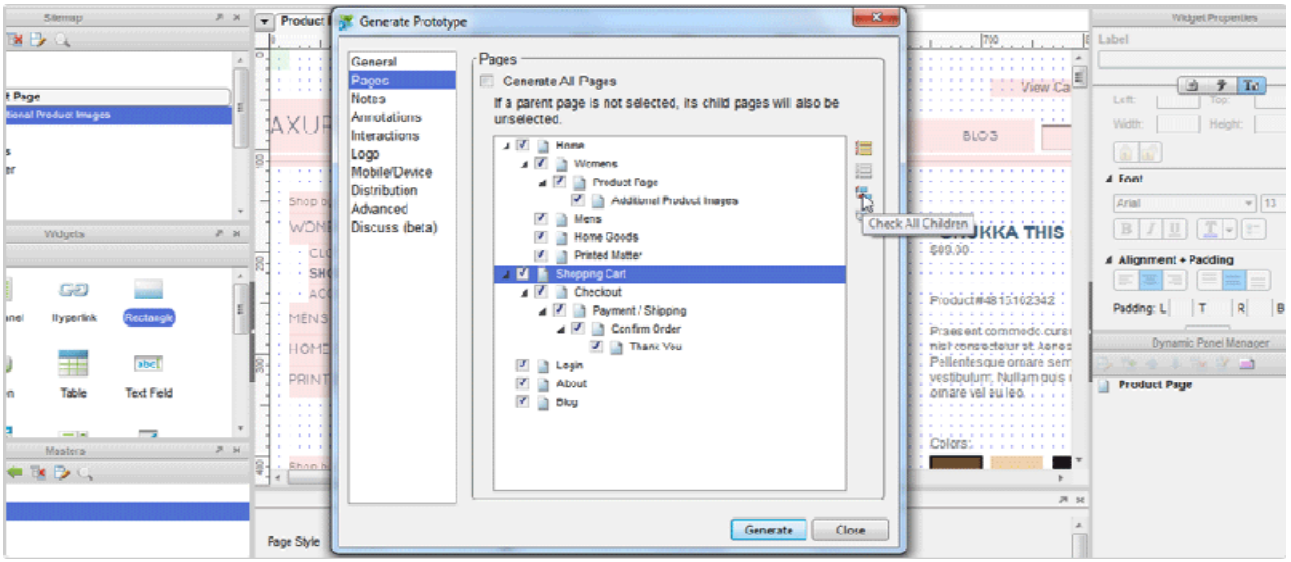


Рисунок 7.17 – Діалогове вікно генерації прототипу

1. **Основні налаштування** (вкладка General). Тут можна вказати каталог, в який буде збережений прототип, і вибрати браузер, за допомогою якого він відкриватиметься після генерації. Прототип складається з декількох файлів, тому для нього варто виділити окремий каталог.
2. **Сторінки** (Pages). Можна вказати, які сторінки мають бути включені в прототип. За замовчуванням встановлена генерація усіх сторінок.
3. **Примітки** (Annotations). Можна вибрати і встановити порядок приміток, які повинні увійти до прототипу.
4. **Взаємодії** (Interactions). Можна вказати, коли повинні відображатися описи сценаріїв.
5. **Логотип** (Logo). Можна завантажити логотип, який представлятиме ваш прототип.
6. **Мобільні пристрої** (Mobile/Device). За допомогою цієї опції можна вставити в прототип тег вікна перегляду. Він вказує прототипу розмір і можливості масштабування вікна в пристрої, через який відкритий прототип.
7. **Поширення** (Distribution, тільки для PC). Опція, яка дозволяє створювати CHM-версію прототипу для його передачі одним файлом.
8. **Розширені налаштування** (Advanced). Опції, що дозволяють зберегти текст як зображення, вибрати одиниці виміру для розмірів шрифтів і відмінити ефект ескізу.
9. **Обговорення** (Discuss, Beta). Опція, що включає можливість обговорення прототипу за допомогою AxShare (share.axure.com). Ця можливість поки що знаходиться на стадії Бета- версії.

Генерація прототипу

Після того, як було налаштовано усі параметри генерації прототипу, натисніть кнопку Generate щоб створити усі файли прототипу, зберегти їх у вказаному каталозі і відкрити прототип у вибраному браузері (рис. 7.18).

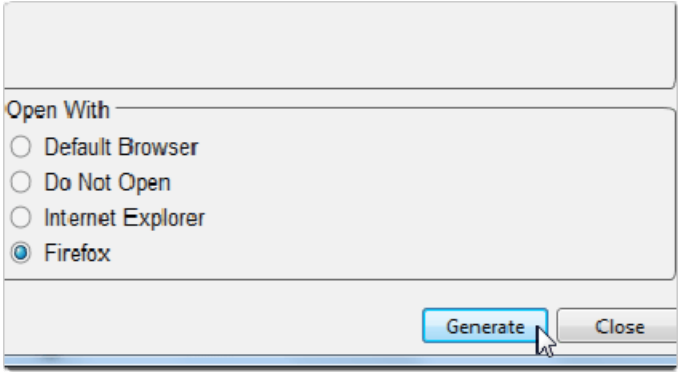


Рисунок 7.18 – Генерація прототипу

Також запустити прототип у будь-який час можна відкривши файл start.html (або index.html) в каталозі з прототипом.

Використання прототипу

Прототип складається з двох частин:

1. Ліворуч розміщується Карта сайту (Sitemap), Замітки (Page Notes), Обговорення (Discuss). Ці опції розташовані в трьох вкладках цього вікна. Карта сайту дозволяє відкрити сторінку в основному фреймі. У вкладці Замітки показані замітки на поточній сторінці. Вкладка Обговорення дозволяє додавати коментарі до прототипу.

Цей фрейм можна згорнути, натиснувши на сірий прямокутник в правому верхньому кутку. Щоб розгорнути фрейм знову, натисніть на сірий прямокутник у верхньому лівому кутку сторінки. Замість того щоб запускати файл start.html, ви можете відразу відкрити html-файл однієї із сторінок прототипу, і тоді лівий фрейм не відображатиметься.

2. Справа розташовується головний фрейм, в якому ви можете переглядати сторінки прототипу, а також діаграми. У цьому вікні усі встановлені вами взаємодії відображаються так, як ви їх задали в Axure RP. Біля віджетів з примітками відображається іконка примітки. Щоб їх прочитати, клікніть на іконку примітки.

Відкритий доступ до прототипу

Є декілька способів відкрити доступ до прототипу, причому для цього користувачам прототипу не обов'язково встановлювати Axure RP або будь-який інший рідер.

Публікація в AxShare (beta)

AxShare – це сервіс для розміщення прототипів (поки що на стадії бета-версії), який дозволяє зберігати до десяти прототипів на одному акаунті. Досить завантажити в AxShare файл RP з прототипом, і ви отримаєте посилання на свій прототип, яке можна надавати користувачам. Прототип також можна захистити паролем. Ця послуга надається безкоштовно.

Також в лівому фреймі є вкладка Discuss, за допомогою якої можна додавати коментарі до прототипу. Усі обговорення ви можете переглянути на сторінці свого акаунта на AxShare.

Пересилка CHM-файлу (тільки для PC)

CHM – це формат HTML-файлу, розроблений Microsoft. На більшості комп'ютерів з Windows є встановлений рідер таких файлів. Як і у випадку з zip-файлом, цей спосіб дозволяє пересилати прототип одним файлом, і, як правило, не вимагає від одержувача установки будь-яких програм для його перегляду.

Щоб створити прототип у форматі .chm, відкрийте вкладку Distribute в діалоговому вікні генерації прототипу (F5), а потім виберіть опцію створення CHM-файлу. При генерації, в каталозі прототипу створиться .chm-файл.

Для створення CHM-файлу повинен бути встановлений HTML Help Workshop. Після його установки знайдіть файл hhc.exe, а потім у вкладі Distribution діалогового вікна генерації прототипу натисніть на Locate hhc.exe і вкажіть шлях до цього файлу, щоб вказати Axure RP розташування програми на вашому комп'ютері.

Завдання на лабораторну роботу

1. Ознайомитися з теоретичними відомостями по роботі.
2. Виконати приклади з уроків розміщених на офіційному сайті:
   - ✔ Show Hidden Panel (http://www.axure.com/learn/dynamic-panels/basic/show-hidden-panel-tutorial)
   - ✔ Tab Control (http://www.axure.com/learn/dynamic-panels/basic/tab-control-tutorial)
   - ✔ Custom Tooltip (http://www.axure.com/learn/dynamic-panels/basic/custom-tooltip-tutorial)
   - ✔ Flyout Menu (http://www.axure.com/learn/dynamic-panels/basic/flyout-menu-tutorial)
3. Виконати публікацію створених прототипів на сервісі AxShare
4. Зберегти створені прототипи у CHM-файл встановивши HTML Help Workshop.
5. Зробити висновки по роботі.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке динамічна панель?
2. Які є стани динамічної панелі?
3. Які особливості використання вікна керування динамічними панелями?
4. Як розробити діалогове вікно для повідомлення про помилку в системі?
5. Яким чином організувати перехід між вікнами програмного продукту?
6. Що таке AxShare

ЛАБОРАТОРНА РОБОТА № 8

Створення прототипу Web-інтерфейсу засобами Axure RP

Мета роботи: виконати аналіз предметної області згідно свого варіанту та навчитися створювати прототипи Web-інтерфейсів за допомогою середовищем проектування і документування графічних інтерфейсів Axure RP.

Теоретичні відомості

8.1 Використання функціональних специфікацій

Функціональні специфікації – це ефективний спосіб для документування, пояснення і узгодження дизайну проекту.

Axure RP створює специфікації у форматі Microsoft Word DOCX. Для створення специфікації не обов'язково мати встановлений Word, але для перегляду або редагування отриманого документу він необхідний.

Існує цілий ряд можливостей для налаштування створюваних характеристик, включаючи вибір сторінок і налаштування шаблону проекту (рис. 8.1). Після того, як ви закінчите з налаштуваннями, необхідно натинути на кнопку Generate щоб створити документ Word із специфікаціями.

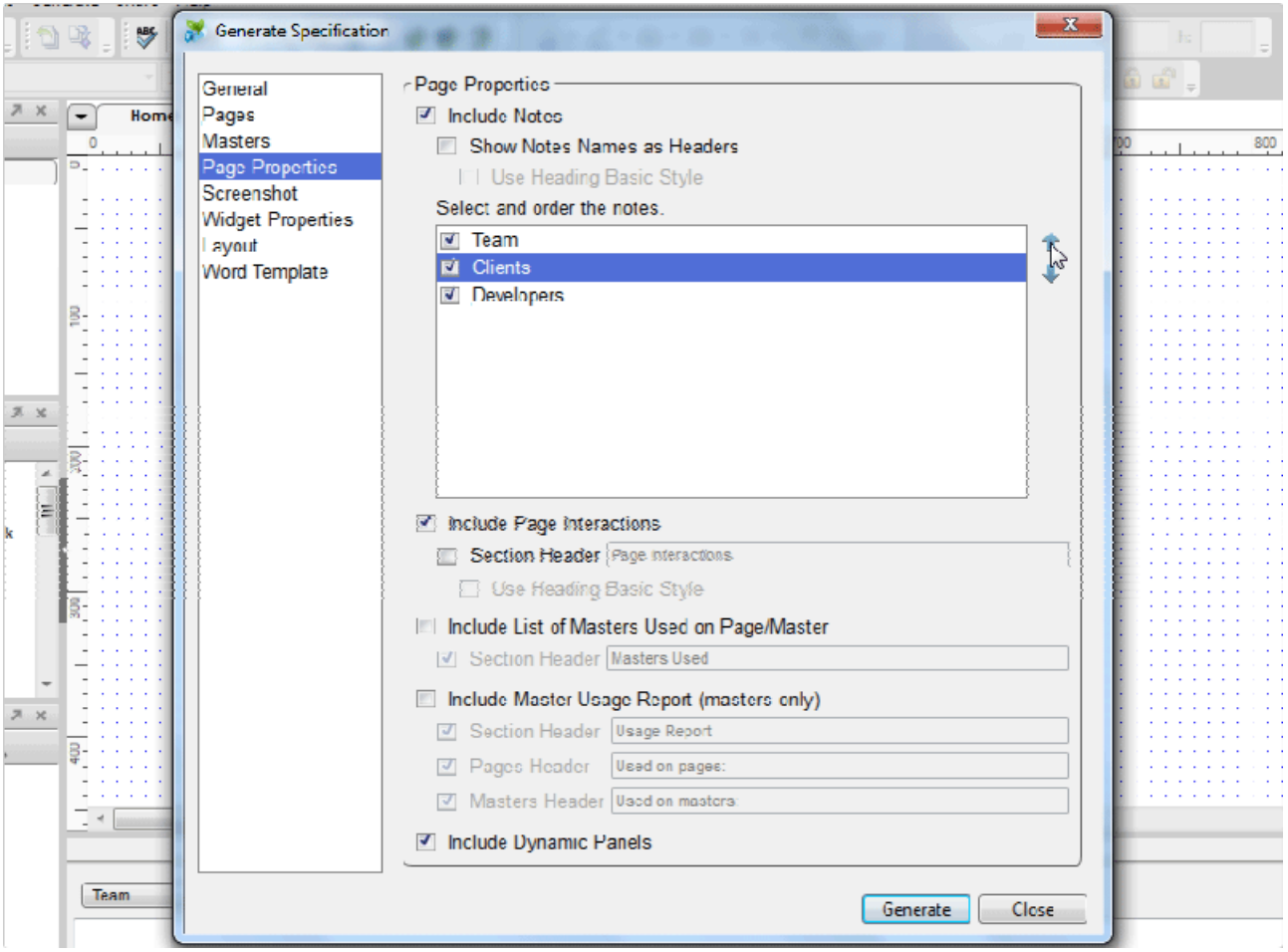


Рисунок 8.1 – Вікно створення файлу специфікацій

Для того, щоб відкрити діалогове вікно створення специфікацій, виберіть в головному меню Generate → Specification або натисніть на панелі інструментів кнопку Specification.

1. **Основні налаштування** (вкладка General). Можна вказати шлях збереження файлу характеристик.
2. **Сторінки** (Pages). Можна вибрати сторінки, які ви хочете включити в специфікацію.
3. **Майстри** (Masters). Можна вибрати майстри, які необхідно включити в специфікацію. Тут же можна знайти опцію, яка дозволяє включати примітки до майстрів.
4. **Властивості сторінок** (Page Properties). Можна вибрати і розташувати у потрібному порядку замітки до сторінок.
5. **Скріншоти** (Screenshots). Опція, яка дозволяє додавати в специфікації скріншоти прототипу.
6. **Властивості віджетів** (Widget Properties). Можна вибрати і розташувати в потрібному порядку примітки до віджетів. Примітки можна розділяти на декілька таблиць.
7. **Компонування** (Layout). Можна вибрати одно- або двох-колонкове компонування відображення специфікації.
8. **Шаблон** (Word Template). Можна вибрати та налаштувати шаблон.

8.2 Налаштування однієї або декількох таблиць опису віджетів

Таблиці віджетів містять описи елементів інтерфейсу (у тому числі міток, ярликів з номерами (footnotes), взаємодій, полів приміток, тексту віджету, тултипів та опції у випадаючих списках). Ці таблиці можна налаштувати у вкладці Widget Properties (рис. 8.2).

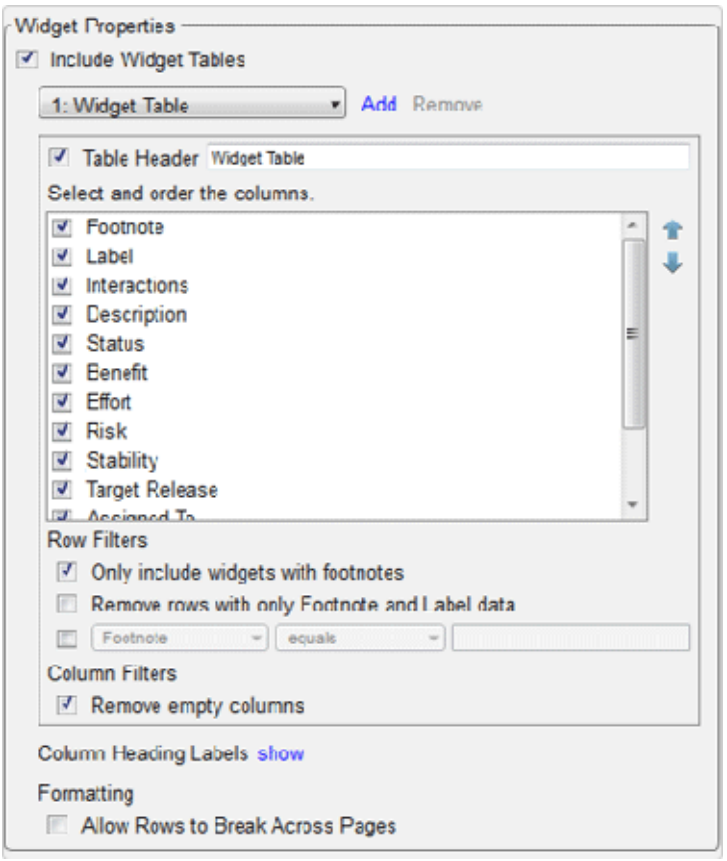


Рисунок 8.2 – Налаштування однієї або декількох таблиць опису віджетів

За замовчуванням в таблицю опису елементів заносяться мітки, ярлики з номерами і поля приміток. Ви можете розширити ці таблиці, щоб розділити

поля з описами на декілька таблиць. Для цього натисніть кнопку Add поряд з випадаючим списком, що містить назви таблиць.

Фільтри і специфікації контенту

Фільтри використовуються для обмеження кількості рядків в таблиці опису віджетів. За замовчуванням в таблицю заносяться тільки ті елементи, які мають ярлики з номерами, що означає, що їм задані примітки або взаємодії.

Якщо ви хочете згенерувати специфікацію контенту, яка міститиме інформацію про текстові віджети, зніміть відмітку з чекбокса «Only include widgets with footnotes» (Включати тільки елементи, що мають ярлики з номерами), і відмітьте чекбокс «Widget Text».

Також можна вибрати опцію «Remove rows with only Footnote and Label data», яка дозволяє видаляти із створюваної таблиці рядки з описом елементів, у яких є тільки ярлики і назви (а усі інші атрибути порожні). Можна налаштувати фільтр на включення в специфікацію описів тільки тих елементів, значення полів яких відповідають заданими вами. Наприклад, «Target Release equals 1.1».

Компонування

У вкладці Layout ви можете налаштувати порядок відображення блоків з інформацією (рис. 8.3). Наприклад, одна таблиця опису віджетів може розміщуватися над скріншотом сторінки, а інша – під ним.

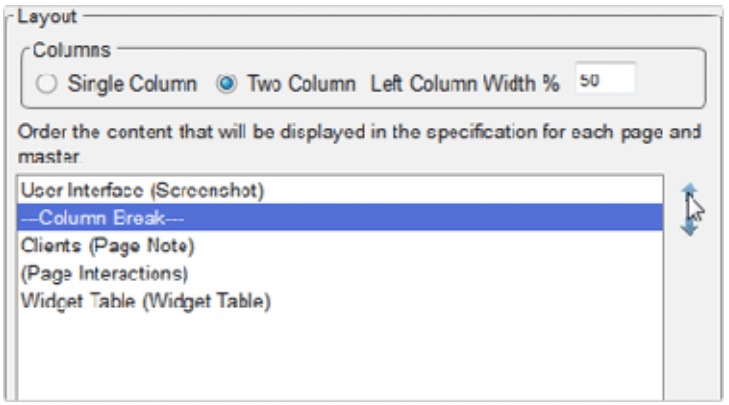


Рисунок 8.3 – Налаштування порядку відображення блоків з інформацією

Також можна вибрати одне- або двох-колонкове компонування. Коли колонок дві, ширину лівої колонки можна вказувати у відсотках (ширина зазвичай налаштовується для того, щоб можна було вставити скріншот великих розмірів). Розмір скріншотів можна налаштувати у вкладці Screenshot (де можна визначити максимальну ширину кожного скріншоту у відсотковому співвідношенні до ширини колонки/сторінки і максимальну висоту у відсотковому співвідношенні до висоти сторінки).

В двох-колонковому компонуванні ліву колонку краще відвести під скріншоти, а праву – для заміток і таблиць опису віджетів.

Користувацькі шаблони Word

У вкладці Word Template ви можете редагувати або імпортувати шаблон, в який буде згенерована функціональна специфікація.

Шаблон – це стандартний файл з розширенням «.docx», який може містити призначені для користувача хедери, футери, заголовок, або будь-який

інший контент, що використовується у документації.

Також у вкладці Word Template можна змінювати стилі документу. У шаблоні має бути присутнім текст [[INSERT AXURE SPEC]], який вказує місце, куди генератор вставить специфікацію.

Для того, щоб відредагувати шаблон, натисніть на кнопку «Edit». Поточний шаблон відкриється для редагування в Microsoft Word. Після того, як ви закінчите редагування,

збережіть файл і закрийте Word. Коли ви зберігаєте файл з розширенням «.rp», шаблон зберігається разом з ним. Можна імпортувати шаблон за допомогою кнопки «Import».

При натисненні кнопки «New Template» відкриється діалогове вікно створення нового шаблону, де можна вибрати розмір сторінки, орієнтацію сторінки і нумерацію заголовків.

Завдання на лабораторну роботу

1. Ознайомитися з теоретичними відомостями по роботі.
2. Розробити графічний прототип Web-інтерфейсу відповідно до основних принципів проектування інтерфейсу. Інтерфейс має бути достатній для виконання усіх сценаріїв поданих у варіантах завдань.
3. Створити функціональну специфікацію розробленого Web-інтерфейсу.
4. Зробити висновки по роботі.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке функціональна специфікація ?
2. Як зробити налаштування однієї або декількох таблиць опису віджетів?
3. Як виконується налаштування порядку відображення блоків з інформацією?
4. Що таке шаблон?

СПИСОК ЛІТЕРАТУРИ

1. Пасєка М. С. Людино-машинний інтерфейс : консп. лек. Івано-Франківськ : ІФНТУНГ, 2015. 194 с.
2. Архангельский А.Я. Программирование в C++ Builder. / Архангельский А.Я. - Бином, 7-е издание, 2010 – 1024 с.
3. Архангельский А.Я. Функции C++, C++ Builder 5 и API Windows / Архангельский А.Я. - М.: ЗАО "Издательство БИНОМ", 2000г. – 760 с.
4. Ткачук М.В. Уніфіковани програмні сервіси та візуальні інтерфейси в інтранет-системах управління технологічними процесами - Системні дослідження та інформаційні технології -№1 - 2004.
5. Глушаков С.В. Программирование на Visual C++ 6.0. / Глушаков С.В., Коваль А.В., Черепнин С.А. - Харків: Фолио, 2002. - 726с.
6. Круглински Д.Дж. Программирование на Microsoft Visual C + + 6.0 для профессионалов. / Круглински Д.Дж., Уингоу С., Дж. Шеферд - СПб: Русская редакция, 2001. - 864 с.
7. C/C++. Программирование на языке высокого уровня: Учебник / Павловская Т.А.. - СПб.: Питер, 2002. - 464с.
8. Логунова О.С. Человеко-машинное взаимодействие: теория и практика Учебное пособие / О.С. Логунова, И. М. Ячиков, Е.А. Ильина. -Ростов н/Д: Феникс, 2006. -285 с.