

1- Date/time localization

1-1- Output date/time:

xoops developers should use formatTimestamp XOOPS function in all output date/times. Using other functions like php date function is invalid.

1-2- Input date/time:

It gets the date from user in the local format and changed it to int number for storing in the database.

It means all developers should use Local input function wherever they get a date/time from user side.

There is no standard function in XOOPS API until now.

There will be a standard function like xoops_strtotime in the future.

The feature request in sf.net:

https://sourceforge.net/tracker/?func=detail&aid=3537303&group_id=41586&atid=430843

until that time developers that want to use strtotime php command encouraged to localize it using the following method.

Replace this:

```
strtotime($inputdate)
```

With this:

```
method_exists('XoopsLocal', 'strtotime') ? XoopsLocal::strtotime($inputdate) :  
strtotime($inputdate)
```

2- Local calendar

"jQuery UI datepicker" is the only calendar that can provide us localization.

Xoops core should implement it in the next 2.6.0 version.

All calendar module developers should use this calendar for the user side.

All calendar module developers should store date/time in int format using the localized input date function in the database.

Feature request here:

https://sourceforge.net/tracker/?func=detail&aid=3405407&group_id=41586&atid=430843

timngno at xoops.org did a nice job here:

<http://xoops.svn.sourceforge.net/viewvc/xoops/XoopsCore/tags/2.6%20timngno/htdocs/class/xoopsform/formdatepicker.php?revision=10091&view=markup>

3- Pdf maker

Only tcpdf is capable to create pdf for local parties.

Xoops core should implement it in the next 2.6.0 version.

Until that time all module developers should call it from

./Frameworks/tcpdf/tcpdf.php

makepdf.php for any module should use tcpdf.

Frank_black at xoops.org work on tcpdf for xoops and the last version is here:

<http://sourceforge.net/projects/chgxoops/files/Frameworks/>

4- LTR -RTL

All themes (or modules in some templates like print or in any iframe) should contain

the following smarty variable in the head wherever they call style and js files:

```
<!-- customized header contents -->
```

```
<{$xoops_module_header}>
```

Any page without the above smarty is invalid.

Xoops core will add this definition in the next 2.6.0 release.

```
define('_RTL',false); //for english and other ltr languages
```

```
define('_RTL',true); //for rtl languages
```

The above is useful for some localization like pdf maker using tcpdf.

the feature request in sf.net here:

https://sourceforge.net/tracker/?func=detail&aid=3555522&group_id=41586&atid=430843

5- Translations:

All language files must end (for now) with CR+LF (windows)

All translations should be UTF-8 without BOM.

All definitions should be free from extra spaces.

example:

Incorrect: `define( '_MD_XFGB_CONFIRM_CODE', 'Confirm verification code' ) ;`

Correct: `define('_MD_XFGB_CONFIRM_CODE','Confirm verification code');`

Module developers should read/use core definitions wherever possible and avoid double definitions and conflict with the core definitions.

All definitions should be plain text and should not contain any format, %s or url like the following example:

```
define("_MD_NEWBB_SEEWAITREPORT","There were <font color=\"red\">
<b>%s</b> Contributions reported </font>");
```

%s and any html will be considered as hardcode and is not allowed.

Any empty definition is not allowed like the following example.

```
define("_MD_PDF_DESC", "");
```

In the future Xoops version there is not any php definitions and core/modules will use INI or Gettext or something else.

feature request in sf.net:

https://sourceforge.net/tracker/?func=detail&aid=3555526&group_id=41586&atid=430843

So please remember these rules:

The source language would be en-US.

All translations should be translated from source en_US language.

If core team choose INI files:

Content of the language file must not be over 150KB

Filenames must not be longer than 50 characters including .ini extension.

There can be plural forms in INI files and this feature must be used

Full customization of language files in themes will be possible like below:

themes/YOUR_THEME/module/YOUR_MODULE/language/YOUR_LANG

6- Images

From this time go on all module developers encouraged to save images in this folder:

Modules/YOUR_MODULE/templates/images

Wherever possible please do not use images with text or having left/right alignment. But if you have these images you should separate images with text or having left/right alignment and move them to language directories.

All other images like icons should be language free (no text or alignment)

Example:

Modules/YOUR_MODULE/templates/images/icons

Modules/YOUR_MODULE/templates/images/language/english

Modules/YOUR_MODULE/templates/images/language/persian

Please pay attention that in the future module developers don't need to do anything for their images. They just define their images in a images.php file and XOOPS will read them.

for example instead of this method:

```

```

you create a smarty variable in your script:

```
$icon_pdf= displayImage('pdf', _PDF);
```

and just use that in your theme/template

```
<{$icon_pdf}>
```

Also full image customization would be possible per theme.

themes/YOUR_THEME/module/YOUR_MODULE/images/icons

themes/YOUR_THEME/module/YOUR_MODULE/images/language/english

themes/YOUR_THEME/module/YOUR_MODULE/images/language/persian

Also custom style.css for modules should be different for each language like this:

Modules/YOUR_MODULE/templates/images/language/english/style.css

Modules/YOUR_MODULE/templates/images/language/persian/style.css

and style.css for each module can be customized for each theme for each language:

themes/YOUR_THEME/module/YOUR_MODULE/images/language/english/style.css

themes/YOUR_THEME/module/YOUR_MODULE/images/language/persian/style.css

Also theme images should follow the same rule.

themes/YOUR_THEME/icons

(now it is like this in 2.6.0 alpha 1: themes/YOUR_THEME/icons) => no problem

themes/YOUR_THEME/img/en

themes/YOUR_THEME/img/fa

(now it is like this in 2.6.0 alpha 1: themes/YOUR_THEME/img) => language needed

And theme style.css encouraged to be in themes language folder:

themes/YOUR_THEME/css/en/style.css

themes/YOUR_THEME/css/fa/style.css

(now it is like this in 2.6.0 alpha 1: themes/YOUR_THEME/css) => language needed

Also the above is just good to have customize style for each language because we have `<{$xoops_module_header}>` for localized style in whole website.

related feature requests here:

https://sourceforge.net/tracker/?func=detail&aid=3545765&group_id=41586&atid=430843

https://sourceforge.net/tracker/?func=detail&aid=3545768&group_id=41586&atid=430843

7- local numbers

Latin based character territories are not familiar with "non-positional numeral systems" in the world but maybe you could understand if you read this wiki:

http://en.wikipedia.org/wiki/List_of_numeral_systems

example roman numeral system:

http://en.wikipedia.org/wiki/Roman_numerals

currently we have a function called `number_format` in php and it is enhanced by xoops previous developers. so developers (core and module) can use it like this:

```
xoops_load('XoopsLocal');  
.....  
XoopsLocal::number_format($VAL)
```

Anyway the priority of this feature is low because it just is a cosmetic and nice looking feature and all non Latin characters countries are familiar with latin 0 1 2 3 4 5 6 7 8 9 numbers.

Just as a technical use we can use it to define local math captcha like this:

1 + 2 = 3 is easy for robots to find.

but IMO

I + II = III (roman numbers) is hard to find for robots.

feature request here:

https://sourceforge.net/tracker/?func=detail&aid=3555533&group_id=41586&atid=430843