Pointer Events implementation outline

http://goo.gl/lg9aYk

Spec: https://dvcs.w3.org/hg/pointerevents/raw-file/tip/pointerEvents.html
Original MSOpenTech WebKit patch may be useful

- **[Done: bug]** Define PointerEvent type (behind RuntimeEnabledFeatures) [S]
    - Add PointerEvent.idl, .h, .cpp
    - Tests that verify you can create / dispatch from JS
    - Add GlobalEventHandlers entries (onpointerdown, etc.) [S]
- **[Done: bug]** Fire pointer events for all touch events [M]
    - blink responsible for converting PlatformTouchEvent into DOM PointerEvents.
    - touch events are fired only from EventHandler::handleTouchEvent
    - we'll have to convert the event-bundle style into separate individual events, but this should be trivial (just walk the changedTouches list, fire a new pointer event for each).
    - fire the touch event only if none of the pointer events are consumed
    - This will require some logic and state for determining isPrimary correctly, but should be easy
- **[bug]** Extend WebMouseEvent to become a pointer event API
    - Eg. add pointer type enum, stylus tilt properties, etc.
- **[bug]** Update MouseEvent (and so PointerEvent) to support fractional co-ordinates [M]
    - At least as much of http://crbug.com/456625 as is necessary to have PointerEvent instances for touch with fractional co-ordinates.
- **[Done: bug]** Add pointer event types to devtools event support [S]
    - Event listener breakpoints, viewing event listeners
    - Updates to monitorEvents(), getEventListeners()
    - TimeLine panel support
    - FilmStripView support
    - Possible future addition: expose effective touch-action?
- **[Done: bug]** Do pointer event dispatch perf testing
    - Eg. does the PE event dispatch logic make 10-finger scenarios (paint/game) at risk for jank on low-end android devices?
    - Perhaps we should have a fast-path to skip creating/dispatching pointer events when we see there are no PE handlers.
- **[Done: bug]** Support sending pointercancel on touch scroll/pinch [S]
    - Use WebTouchEvent::causesScrollingIfUncanceled - when true, send pointercancel and stop creating pointer events for touch events
- **[Done: bug]** Fire pointer events for all mouse events [L]

    - Find all code (other than tap) that dispatches a mouse* event, and ensure it first dispatches the appropriate pointer event, sending the mouse event only when the pointer event is uncancelled
    - There is a lot of scattered and brittle code firing mouse events (eg. synthetic mousemove events) - we'll need to collect it all, probably clean some things up to be more manageable.
    - Note that we'll need to be careful to ensure any default actions are executed exactly once (not once for the pointer event and once for the mouse event).
- **[Done: bug]** Add chorded button transformation to mouse->pointer event generation logic [S]
    - Should be relatively easy. See http://www.w3.org/TR/pointerevents/#chorded-button-interactions
- **[Done]** Implement new direction-specific touch-action values [S]
    - Wait for standard to be updated
    - Can be shipped independently from the rest of PE
    - Gnana at Samsung a good candidate to implement (he did some of the CLs for pan-x/pan-y)
- **[Done]** Implement pointer capture support [L]
    - This may require cleaning up handling of mouse events throughout blink, unifying logic and removing assumptions around targeting behavior
- Implement touch-action hit testing in CC [M]
    - This can be done in parallel (ideall by someone with compositing experience), and is something we want for scroll-blocks-on anyway
    - After discussion with aelias we think we should just block this work on slimming paint.
    - Will need cc-side reviewer, probably aelias
- **[Done]** Add pointer event handler tracking [M]
    - using EventHandlerRegistry
    - plumb 'has pointer event handlers' back to Chrome's InputRouter
    - If there are pointer event handlers but NOT touch event handlers (or handlers only outside the region touched) then we send the events async (like scroll-blocks-on: none).
    - Will involve some small changes to content and cc input handling - reviewers: jdduke/aelias
- Note: Everything below this point could be done in a phase 2 - post ship
- Rename WebMouseEvent to WebPointerEvent
    - might as well block on the chromium/blink merge when this will be much easier
- Consider consuming OS pointer events on Windows 8 for maximum compat with IE [M]
    - what else would this buy us? Ok to drop touch support on Win7 and below?
- Implement support for stylus by generating WebPointerEvent on each platform: [L]
    - Android [reviewer: jdduke]
    - Windows [reviewer: ? sadrul, sky]
    - Mac [reviewer: ccameron?]

    - ChromeOS (ozone and X11) [reviewer: sadrul]
    - Linux X11 [reviewer: sadrul]
- Consider changing chromium to generate just WebPointerEvents, remove WebMouseEvent and WebTouchEvent [L]

Some issues to consider:

- We need touch-action hit-testing on the impl thread, but we want this for scroll-blocks-on anyway.
    - Slimming paint should make this easy, probably need to block shipping PE on slimming paint
    - The alternative of promoting layers with touch-action is likely not acceptable from a perf perspective in some edge cases
    - Another option is to implement a region-tracking style system (like cc touch hit testing), but if that would be throw-away work then we're better off to wait for slimming paint.
- We'll need to support sending pointer events async without blocking scrolling (for when there's a PE handler but not a TE handler). Again outstanding scroll-blocks-on work requires this anyway (it's basically the same as scroll-blocks-on: none), but it's a little involved.
- The mouse event handling code in EventHandler is old and notoriously brittle. We'll probably need to clean this up a bit to properly separate out correct default action handling from event generation. Mouse event generation will become optional (based on non-consumption of pointer events) but the default action needs to still apply exactly once.
- Plumbing proper stylus support will require platform-specific changes in Chromium and probably a new WebPointerEvent type (which we could someday migrate touch and move over to as well). But this can probably come later - ship just mouse/touch first.
- Do pointer capture APIs affect touch event targetting? Ideally I think we'd avoid needing two separate event paths for the different event types.
- Adding the explicit capture APIs (setPointerCapture) may break some assumptions in the existing mouse handling code (but should be easy for touch). Again, some significant EventHandler cleanup and bug-tail-chasing may be required.