blob: 6fd3cad44dd0212d4c589274ce2b91841b7dca8d [file] [log] [blame] [edit]

```
// Copyright (c) 2012 The Chromium Authors. All rights reserved.
// Use of this source code is governed by a BSD-style license that can be
// found in the LICENSE file.

#include "content/public/common/content_switches.h"

namespace switches {

// By default, file:// URIs cannot read other file:// URIs. This is an
// override for developers who need the old behavior for testing.
const char kAllowFileAccessFromFiles[]      = "allow-file-access-from-files";

// Allows debugging of sandboxed processes (see zygote_main_linux.cc).
const char kAllowSandboxDebugging[]         = "allow-sandbox-debugging";

// Allow compositing on chrome:// pages.
const char kAllowWebUICompositing[]         = "allow-webui-compositing";

// Enumerates and prints a child process' most dangerous handles when it
// is terminated.
const char kAuditHandles[]                  = "enable-handle-auditing";

// The same as kAuditHandles except all handles are enumerated.
const char kAuditAllHandles[]               = "enable-handle-auditing-all";

// Causes the browser process to throw an assertion on startup.
const char kBrowserAssertTest[]             = "assert-test";

// Causes the browser process to crash on startup.
const char kBrowserCrashTest[]              = "crash-test";

// Path to the exe to run for the renderer and plugin subprocesses.
const char kBrowserSubprocessPath[]         = "browser-subprocess-path";

// Run Chrome in Chrome Frame mode. This means that Chrome expects to be run
// as a dependent process of the Chrome Frame plugin.
const char kChromeFrame[]                   = "chrome-frame";

// Disables client-visible 3D APIs, in particular WebGL and Pepper 3D.
// This is controlled by policy and is kept separate from the other
// enable/disable switches to avoid accidentally regressing the policy
// support for controlling access to these APIs.
const char kDisable3DAPIs[]                 = "disable-3d-apis";

// Disable gpu-accelerated 2d canvas.
const char kDisableAccelerated2dCanvas[]    = "disable-accelerated-2d-canvas";

// Disables accelerated compositing.
const char kDisableAcceleratedCompositing[] = "disable-accelerated-compositing";

// Disables the hardware acceleration of 3D CSS and animation.
const char kDisableAcceleratedLayers[]      = "disable-accelerated-layers";

// Disables the hardware acceleration of plugins.
const char kDisableAcceleratedPlugins[]     = "disable-accelerated-plugins";

// Disables GPU accelerated video display.
const char kDisableAcceleratedVideo[]       = "disable-accelerated-video";

// Disables the alternate window station for the renderer.
const char kDisableAltWinstation[]          = "disable-winsta";

```

```
63  // Disable the ApplicationCache.
64  const char kDisableApplicationCache[]       = "disable-application-cache";
65  //
66  // TODO(scherkus): remove --disable-audio when we have a proper fallback
67  // mechanism.
68  const char kDisableAudio[]                  = "disable-audio";
69
70  // Disable limits on the number of backing stores. Can prevent blinking for
71  // users with many windows/tabs and lots of memory.
72  const char kDisableBackingStoreLimit[]      = "disable-backing-store-limit";
73
74  // Disables HTML5 DB support.
75  const char kDisableDatabases[]              = "disable-databases";
76
77  // Disables data transfer items.
78  const char kDisableDataTransferItems[]      = "disable-data-transfer-items";
79
80  // Disable deferred 2d canvas rendering.
81  const char kDisableDeferred2dCanvas[]        = "disable-deferred-2d-canvas";
82
83  // Disables desktop notifications (default enabled on windows).
84  const char kDisableDesktopNotifications[]   = "disable-desktop-notifications";
85
86  // Disables device orientation events.
87  const char kDisableDeviceOrientation[]      = "disable-device-orientation";
88
89  // Disable experimental WebGL support.
90  const char kDisableExperimentalWebGL[]      = "disable-webgl";
91
92  // Blacklist the GPU for accelerated compositing.
93  const char kBlacklistAcceleratedCompositing[] =
94      "blacklist-accelerated-compositing";
95
96  // Blacklist the GPU for WebGL.
97  const char kBlacklistWebGL[]                = "blacklist-webgl";
98
99  // Disable FileSystem API.
100 const char kDisableFileSystem[]             = "disable-file-system";
101
102 // Disable 3D inside of flapper.
103 const char kDisableFlash3d[]                = "disable-flash-3d";
104
105 // Disable Stage3D inside of flapper.
106 const char kDisableFlashStage3d[]           = "disable-flash-stage3d";
107
108 // Suppresses support for the Geolocation javascript API.
109 const char kDisableGeolocation[]            = "disable-geolocation";
110
111 // Disable GL multisampling.
112 const char kDisableGLMultisampling[]        = "disable-gl-multisampling";
113
114 // Do not launch the GPU process shortly after browser process launch. Instead
115 // launch it when it is first needed.
116 const char kDisableGpuProcessPrelaunch[]    = "diasable-gpu-process-prelaunch";
117
118 // Disable the GPU process sandbox.
119 const char kDisableGpuSandbox[]             = "disable-gpu-sandbox";
120
121 // Reduces the GPU process sandbox to be less strict.
122 const char kReduceGpuSandbox[]              = "reduce-gpu-sandbox";
123
124 // Enable the GPU process sandbox (Linux/Chrome OS only for now).
125 const char kEnableGpuSandbox[]              = "enable-gpu-sandbox";
126
127 // Suppresses hang monitor dialogs in renderer processes.  This may allow slow
128 // unload handlers on a page to prevent the tab from closing, but the Task
129 // Manager can be used to terminate the offending process in this case.
130 const char kDisableHangMonitor[]            = "disable-hang-monitor";
131
132 // Disable the use of an ImageTransportSurface. This means the GPU process
```

```
133 // will present the rendered page rather than the browser process.
134 const char kDisableImageTransportSurface[]  = "disable-image-transport-surface";
135
136 // Disables GPU hardware acceleration.  If software renderer is not in place,
137 // then the GPU process won't launch.
138 const char kDisableGpu[]                    = "disable-gpu";
139
140 // Disable the thread that crashes the GPU process if it stops responding to
141 // messages.
142 const char kDisableGpuWatchdog[]            = "disable-gpu-watchdog";
143
144 // Prevent Java from running.
145 const char kDisableJava[]                   = "disable-java";
146
147 // Don't execute JavaScript (browser JS like the new tab page still runs).
148 const char kDisableJavaScript[]             = "disable-javascript";
149
150 // Disable JavaScript I18N API.
151 const char kDisableJavaScriptI18NAPI[]      = "disable-javascript-i18n-api";
152
153 // Disable LocalStorage.
154 const char kDisableLocalStorage[]           = "disable-local-storage";
155
156 // Force logging to be disabled.  Logging is enabled by default in debug
157 // builds.
158 const char kDisableLogging[]                = "disable-logging";
159
160 // Prevent plugins from running.
161 const char kDisablePlugins[]                = "disable-plugins";
162
163 // Disable the JavaScript Pointer Lock API.
164 const char kDisablePointerLock[]            = "disable-pointer-lock";
165
166 // Disable pop-up blocking.
167 const char kDisablePopupBlocking[]          = "disable-popup-blocking";
168
169 // Disables remote web font support. SVG font should always work whether this
170 // option is specified or not.
171 const char kDisableRemoteFonts[]            = "disable-remote-fonts";
172
173 // Turns off the accessibility in the renderer.
174 const char kDisableRendererAccessibility[]  = "disable-renderer-accessibility";
175
176 // Disable False Start in SSL and TLS connections.
177 const char kDisableSSLFalseStart[]          = "disable-ssl-false-start";
178
179 // Disable smooth scrolling for testing.
180 const char kDisableSmoothScrolling[]        = "disable-smooth-scrolling";
181
182 // Disable the seccomp sandbox (Linux only)
183 const char kDisableSeccompSandbox[]         = "disable-seccomp-sandbox";
184
185 // Disable the seccomp filter sandbox (Linux only)
186 const char kDisableSeccompFilterSandbox[]   = "disable-seccomp-filter-sandbox";
187
188 // Disable session storage.
189 const char kDisableSessionStorage[]         = "disable-session-storage";
190
191 // Enable shared workers. Functionality not yet complete.
192 const char kDisableSharedWorkers[]          = "disable-shared-workers";
193
194 // Disables site-specific tailoring to compatibility issues in WebKit.
195 const char kDisableSiteSpecificQuirks[]     = "disable-site-specific-quirks";
196
197 // Disables speech input.
198 const char kDisableSpeechInput[]            = "disable-speech-input";
199
200 // Enables scripted speech api.
201 const char kEnableScriptedSpeech[]          = "enable-scripted-speech";
202
```

```
203 // TODO(primiano): Remove the two switches below when the URL becomes public.
204 // Specifies the webservice URL for continuous speech recognition.
205 const char kSpeechRecognitionWebserviceURL[] = "speech-service";
206
207 // Specifies the request key for the continuous speech recognition webservice.
208 const char kSpeechRecognitionWebserviceKey[] = "speech-service-key";
209
210 // Disables animation on the compositor thread.
211 const char kDisableThreadedAnimation[]      = "disable-threaded-animation";
212
213 // Disable web audio API.
214 const char kDisableWebAudio[]               = "disable-webaudio";
215
216 // Don't enforce the same-origin policy. (Used by people testing their sites.)
217 const char kDisableWebSecurity[]            = "disable-web-security";
218
219 // Disable Web Sockets support.
220 const char kDisableWebSockets[]             = "disable-web-sockets";
221
222 // Disables WebKit's XSSAuditor. The XSSAuditor mitigates reflective XSS.
223 const char kDisableXSSAuditor[]             = "disable-xss-auditor";
224
225 // Specifies if the |DOMAutomationController| needs to be bound in the
226 // renderer. This binding happens on per-frame basis and hence can potentially
227 // be a performance bottleneck. One should only enable it when automating dom
228 // based tests. Also enables sending/receiving renderer automation messages
229 // through the |AutomationRenderViewHelper|.
230 //
231 // TODO(kkania): Rename this to enable-renderer-automation after moving the
232 // |DOMAutomationController| to the |AutomationRenderViewHelper|.
233 const char kDomAutomationController[]       = "dom-automation";
234
235 // Enable hardware accelerated page painting.
236 const char kEnableAcceleratedPainting[]     = "enable-accelerated-painting";
237
238 // Enable gpu-accelerated SVG/W3C filters.
239 const char kEnableAcceleratedFilters[]      = "enable-accelerated-filters";
240
241 // Turns on extremely verbose logging of accessibility events.
242 const char kEnableAccessibilityLogging[]    = "enable-accessibility-logging";
243
244 // Enables the creation of compositing layers for fixed position elements.
245 const char kEnableCompositingForFixedPosition[] =
246     "enable-fixed-position-compositing";
247
248 // Enables CSS3 regions
249 const char kEnableCssRegions[]              = "enable-css-regions";
250
251 // Enables CSS3 custom filters
252 const char kEnableCssShaders[]              = "enable-css-shaders";
253
254 // Enables device motion events.
255 const char kEnableDeviceMotion[]            = "enable-device-motion";
256
257 // Enables support for encrypted media. Current implementation is
258 // incomplete and this flag is used for development and testing.
259 const char kEnableEncryptedMedia[]          = "enable-encrypted-media";
260
261 // Enables the fastback page cache.
262 const char kEnableFastback[]                = "enable-fastback";
263
264 // By default, a page is laid out to fill the entire width of the window.
265 // This flag fixes the layout of the page to a default of 980 CSS pixels,
266 // or to a specified width and height using --enable-fixed-layout=w,h
267 const char kEnableFixedLayout[]             = "enable-fixed-layout";
268
269 // Enable the JavaScript Full Screen API.
270 const char kDisableFullScreen[]             = "disable-fullscreen";
271
272 // Enable the JavaScript Pointer Lock API.
```

```
const char kEnablePointerLock[]             = "enable-pointer-lock";

// Enable the Gamepad API
const char kEnableGamepad[]                 = "enable-gamepad";

// Enables the GPU benchmarking extension
const char kEnableGpuBenchmarking[]         = "enable-gpu-benchmarking";

// Force logging to be enabled.  Logging is disabled by default in release
// builds.
const char kEnableLogging[]                 = "enable-logging";

// Enables Media Source API on <audio>/<video> elements.
const char kEnableMediaSource[]             = "enable-media-source";

// Enable media stream in WebKit.
// http://www.whatwg.org/specs/web-apps/current-work/multipage/dnd.html#mediastream
const char kEnablePeerConnection[]          = "enable-peer-connection";

// On Windows, converts the page to the currently-installed monitor profile.
// This does NOT enable color management for images. The source is still
// assumed to be sRGB.
const char kEnableMonitorProfile[]          = "enable-monitor-profile";

// Enables partial swaps in the WK compositor on platforms that support it.
const char kEnablePartialSwap[]             = "enable-partial-swap";

// Enables touch-screen pinch gestures.
const char kEnablePinch[]                   = "enable-pinch";

// Enable caching of pre-parsed JS script data.  See http://crbug.com/32407.
const char kEnablePreparsedJsCaching[]      = "enable-preparsed-js-caching";

// Enable privileged WebGL extensions; without this switch such extensions are
// available only to Chrome extensions.
const char kEnablePrivilegedWebGLExtensions[] =
    "enable-privileged-webgl-extensions";

// Aggressively free GPU command buffers belonging to hidden tabs.
const char kEnablePruneGpuCommandBuffers[] =
    "enable-prune-gpu-command-buffers";

// Enable renderer side mixing and low latency audio path for media elements.
const char kEnableRendererSideMixing[] = "enable-renderer-side-mixing";

// Enables TLS cached info extension.
const char kEnableSSLCachedInfo[]  = "enable-ssl-cached-info";

// Cause the OS X sandbox write to syslog every time an access to a resource
// is denied by the sandbox.
const char kEnableSandboxLogging[]          = "enable-sandbox-logging";

// Enable the seccomp sandbox (Linux only)
const char kEnableSeccompSandbox[]          = "enable-seccomp-sandbox";

// Enable shadow DOM API
const char kEnableShadowDOM[]          = "enable-shadow-dom";

// Enable <style scoped>
const char kEnableStyleScoped[]             = "enable-style-scoped";

// On platforms that support it, enables smooth scroll animation.
const char kEnableSmoothScrolling[]         = "enable-smooth-scrolling";

// Enables StatsTable, logging statistics to a global named shared memory table.
const char kEnableStatsTable[]              = "enable-stats-table";

// Experimentally ensures that each renderer process:
// 1) Only handles rendering for a single page.
// (Note that a page can reference content from multiple origins due to images,
```

```
// iframes, etc).
// 2) Only has authority to see or use cookies for the page's top-level origin.
// (So if a.com iframe's b.com, the b.com network request will be sent without
// cookies).
// This is expected to break compatibility with many pages for now.
const char kEnableStrictSiteIsolation[]     = "enable-strict-site-isolation";

// Enable multithreaded GPU compositing of web content.
const char kEnableThreadedCompositing[]     = "enable-threaded-compositing";

// Disable multithreaded GPU compositing of web content.
const char kDisableThreadedCompositing[]     = "disable-threaded-compositing";

// Enable use of experimental TCP sockets API for sending data in the
// SYN packet.
const char kEnableTcpFastOpen[]              = "enable-tcp-fastopen";

// Enables hardware acceleration of video decode, where available.
const char kEnableAcceleratedVideoDecode[] = "enable-accelerated-video-decode";

// Enables support for video tracks. Current implementation is
// incomplete and this flag is used for development and testing.
const char kEnableVideoTrack[]               = "enable-video-track";

// Enables the use of the viewport meta tag, which allows
// pages to control aspects of their own layout. This also turns on touch-screen
// pinch gestures.
const char kEnableViewport[]                 = "enable-viewport";

// Enables experimental features for the geolocation API.
// Current features:
// - CoreLocation support for Mac OS X 10.6
// - Gateway location for Linux and Windows
// - Location platform support for Windows 7
const char kExperimentalLocationFeatures[]  = "experimental-location-features";

// Load NPAPI plugins from the specified directory.
const char kExtraPluginDir[]                 = "extra-plugin-dir";

// If accelerated compositing is supported, always enter compositing mode for
// the base layer even when compositing is not strictly required.
const char kForceCompositingMode[]           = "force-compositing-mode";

// This flag disables force compositing mode and prevents it from being enabled
// via field trials.
const char kDisableForceCompositingMode[]    = "disable-force-compositing-mode";

// Some field trials may be randomized in the browser, and the randomly selected
// outcome needs to be propagated to the renderer. For instance, this is used
// to modify histograms recorded in the renderer, or to get the renderer to
// also set of its state (initialize, or not initialize components) to match the
// experiment(s). The option is also useful for forcing field trials when
// testing changes locally. The argument is a list of name and value pairs,
// separated by slashes. See FieldTrialList::CreateTrialsFromString() in
// field_trial.h for details.
const char kForceFieldTrials[]               = "force-fieldtrials";

// Force renderer accessibility to be on instead of enabling it on demand when
// a screen reader is detected. The disable-renderer-accessibility switch
// overrides this if present.
const char kForceRendererAccessibility[]     = "force-renderer-accessibility";

// Passes gpu device_id from browser process to GPU process.
const char kGpuDeviceID[]                    = "gpu-device-id";

// Passes gpu driver_vendor from browser process to GPU process.
const char kGpuDriverVendor[]                = "gpu-driver-vendor";

// Passes gpu driver_version from browser process to GPU process.
const char kGpuDriverVersion[]               = "gpu-driver-version";
```

```
413
414 // Extra command line options for launching the GPU process (normally used
415 // for debugging). Use like renderer-cmd-prefix.
416 const char kGpuLauncher[]                   = "gpu-launcher";
417
418 // Makes this process a GPU sub-process.
419 const char kGpuProcess[]                    = "gpu-process";
420
421 // Causes the GPU process to display a dialog on launch.
422 const char kGpuStartupDialog[]              = "gpu-startup-dialog";
423
424 // Passes gpu vendor_id from browser process to GPU process.
425 const char kGpuVendorID[]                   = "gpu-vendor-id";
426
427 // Used in conjunction with kRendererProcess. This causes the process
428 // to run as a guest renderer instead of a regular renderer.
429 const char kGuestRenderer[]                 = "guest-renderer";
430
431 // Run the GPU process as a thread in the browser process.
432 const char kInProcessGPU[]                  = "in-process-gpu";
433
434 // Runs plugins inside the renderer process
435 const char kInProcessPlugins[]              = "in-process-plugins";
436
437 // Runs WebGL inside the renderer process.
438 const char kInProcessWebGL[]                = "in-process-webgl";
439
440 // Specifies the flags passed to JS engine
441 const char kJavaScriptFlags[]               = "js-flags";
442
443 // Load an NPAPI plugin from the specified path.
444 const char kLoadPlugin[]                    = "load-plugin";
445
446 // Sets the minimum log level. Valid values are from 0 to 3:
447 // INFO = 0, WARNING = 1, LOG_ERROR = 2, LOG_FATAL = 3.
448 const char kLoggingLevel[]                  = "log-level";
449
450 // Make plugin processes log their sent and received messages to VLOG(1).
451 const char kLogPluginMessages[]             = "log-plugin-messages";
452
453 // Causes the process to run as a NativeClient broker
454 // (used for launching NaCl loader processes on 64-bit Windows).
455 const char kNaClBrokerProcess[]             = "nacl-broker";
456
457 // Causes the process to run as a NativeClient loader.
458 const char kNaClLoaderProcess[]             = "nacl-loader";
459
460 // Don't send HTTP-Referer headers.
461 const char kNoReferrers[]                   = "no-referrers";
462
463 // Disables the sandbox for all process types that are normally sandboxed.
464 const char kNoSandbox[]                     = "no-sandbox";
465
466 // Specifies a command that should be used to launch the plugin process.  Useful
467 // for running the plugin process through purify or quantify.  Ex:
468 //   --plugin-launcher="path\to\purify /Run=yes"
469 const char kPluginLauncher[]                = "plugin-launcher";
470
471 // Tells the plugin process the path of the plugin to load
472 const char kPluginPath[]                    = "plugin-path";
473
474 // Causes the process to run as a plugin subprocess.
475 const char kPluginProcess[]                 = "plugin";
476
477 // Causes the plugin process to display a dialog on launch.
478 const char kPluginStartupDialog[]           = "plugin-startup-dialog";
479
480 // Argument to the process type that indicates a PPAPI broker process type.
481 const char kPpapiBrokerProcess[]            = "ppapi-broker";
482
```

```
// Runs PPAPI (Pepper) plugins out-of-process.
const char kPpapiOutOfProcess[]             = "ppapi-out-of-process";

// Like kPluginLauncher for PPAPI plugins.
const char kPpapiPluginLauncher[]           = "ppapi-plugin-launcher";

// Argument to the process type that indicates a PPAPI plugin process type.
const char kPpapiPluginProcess[]            = "ppapi";

// Causes the PPAPI sub process to display a dialog on launch.
const char kPpapiStartupDialog[]            = "ppapi-startup-dialog";

// Runs a single process for each site (i.e., group of pages from the same
// registered domain) the user visits.  We default to using a renderer process
// for each site instance (i.e., group of pages from the same registered
// domain with script connections to each other).
const char kProcessPerSite[]                = "process-per-site";

// Runs each set of script-connected tabs (i.e., a BrowsingInstance) in its own
// renderer process.  We default to using a renderer process for each
// site instance (i.e., group of pages from the same registered domain with
// script connections to each other).
const char kProcessPerTab[]                 = "process-per-tab";

// The value of this switch determines whether the process is started as a
// renderer or plugin host.  If it's empty, it's the browser.
const char kProcessType[]                   = "type";

// Register Pepper plugins (see pepper_plugin_registry.cc for its format).
const char kRegisterPepperPlugins[]         = "register-pepper-plugins";

// Enables remote debug over HTTP on the specified port.
const char kRemoteDebuggingPort[]           = "remote-debugging-port";

// Causes the renderer process to throw an assertion on launch.
const char kRendererAssertTest[]            = "renderer-assert-test";

#if defined(OS_POSIX)
// Causes the renderer process to cleanly exit via calling exit().
const char kRendererCleanExit[]             = "renderer-clean-exit";
#endif

// On POSIX only: the contents of this flag are prepended to the renderer
// command line. Useful values might be "valgrind" or "xterm -e gdb --args".
const char kRendererCmdPrefix[]             = "renderer-cmd-prefix";

// Causes the process to run as renderer instead of as browser.
const char kRendererProcess[]               = "renderer";

// Overrides the default/calculated limit to the number of renderer processes.
// Very high values for this setting can lead to high memory/resource usage
// or instability.
const char kRendererProcessLimit[]          = "renderer-process-limit";

// Causes the renderer process to display a dialog on launch.
const char kRendererStartupDialog[]         = "renderer-startup-dialog";

// Causes the process to run as a service process.
const char kServiceProcess[]                = "service";

// Renders a border around composited Render Layers to help debug and study
// layer compositing.
const char kShowCompositedLayerBorders[]    = "show-composited-layer-borders";

// Draws a textual dump of the compositor layer tree to help debug and study
// layer compositing.
const char kShowCompositedLayerTree[]       = "show-composited-layer-tree";

// Draws a FPS indicator
const char kShowFPSCounter[]                = "show-fps-counter";
```

```
553
554 // Visibly render a border around paint rects in the web page to help debug
555 // and study painting behavior.
556 const char kShowPaintRects[]                = "show-paint-rects";
557
558 // Runs the renderer and plugins in the same process as the browser
559 const char kSingleProcess[]                 = "single-process";
560
561 // Skip gpu info collection, blacklist loading, and blacklist auto-update
562 // scheduling at browser startup time.
563 // Therefore, all GPU features are available, and about:gpu page shows empty
564 // content. The switch is intended only for tests.
565 const char kSkipGpuDataLoading[]            = "skip-gpu-data-loading";
566
567 // Runs the security test for the renderer sandbox.
568 const char kTestSandbox[]                   = "test-sandbox";
569
570 // Causes TRACE_EVENT flags to be recorded from startup. Optionally, can
571 // specify the specific trace categories to include (e.g.
572 // --trace-startup=base,net) otherwise, all events are recorded. Setting this
573 // flag results in the first call to BeginTracing() to receive all trace events
574 // since startup. In Chrome, you may find --trace-startup-file and
575 // --trace-startup-duration to control the auto-saving of the trace (not
576 // supported in the base-only TraceLog component).
577 const char kTraceStartup[]                  = "trace-startup";
578
579 // If supplied, sets the file which startup tracing will be stored into, if
580 // omitted the default will be used "chrometrace.log" in the current directory.
581 // Has no effect unless --trace-startup is also supplied.
582 // Example: --trace-startup --trace-startup-file=/tmp/trace_event.log
583 // As a special case, can be set to 'none' - this disables automatically saving
584 // the result to a file and the first manually recorded trace will then receive
585 // all events since startup.
586 const char kTraceStartupFile[]              = "trace-startup-file";
587
588 // Sets the time in seconds until startup tracing ends. If omitted a default of
589 // 5 seconds is used. Has no effect without --trace-startup, or if
590 // --startup-trace-file=none was supplied.
591 const char kTraceStartupDuration[]          = "trace-startup-duration";
592
593 // Prioritizes the UI's command stream in the GPU process
594 extern const char kUIPrioritizeInGpuProcess[] =
595     "ui-prioritize-in-gpu-process";
596
597 // A string used to override the default user agent with a custom one.
598 const char kUserAgent[]                     = "user-agent";
599
600 // On POSIX only: the contents of this flag are prepended to the utility
601 // process command line. Useful values might be "valgrind" or "xterm -e gdb
602 // --args".
603 const char kUtilityCmdPrefix[]              = "utility-cmd-prefix";
604
605 // Causes the process to run as a utility subprocess.
606 const char kUtilityProcess[]                = "utility";
607
608 // The utility process is sandboxed, with access to one directory. This flag
609 // specifies the directory that can be accessed.
610 const char kUtilityProcessAllowedDir[]      = "utility-allowed-dir";
611
612 // Will add kWaitForDebugger to every child processes. If a value is passed, it
613 // will be used as a filter to determine if the child process should have the
614 // kWaitForDebugger flag passed on or not.
615 const char kWaitForDebuggerChildren[]       = "wait-for-debugger-children";
616
617 // Choose which logging channels in WebCore to activate.  See
618 // Logging.cpp in WebKit's WebCore for a list of available channels.
619 const char kWebCoreLogChannels[]            = "webcore-log-channels";
620
621 // Causes the worker process allocation to use as many processes as cores.
622 const char kWebWorkerProcessPerCore[]       = "web-worker-process-per-core";
```

```
623
624 // Causes workers to run together in one process, depending on their domains.
625 // Note this is duplicated in webworkerclient_impl.cc
626 const char kWebWorkerShareProcesses[]       = "web-worker-share-processes";
627
628 // Causes the process to run as a worker subprocess.
629 const char kWorkerProcess[]                 = "worker";
630
631 // The prefix used when starting the zygote process. (i.e. 'gdb --args')
632 const char kZygoteCmdPrefix[]               = "zygote-cmd-prefix";
633
634 // Causes the process to run as a renderer zygote.
635 const char kZygoteProcess[]                 = "zygote";
636
637 // Enables moving cursor by word in visual order.
638 const char kEnableVisualWordMovement[]      = "enable-visual-word-movement";
639
640 #if defined(OS_POSIX) && !defined(OS_MACOSX)
641 // Specify the amount the trackpad should scroll by.
642 const char kScrollPixels[]                  = "scroll-pixels";
643 #endif
644
645 #if defined(OS_MACOSX) || defined(OS_WIN)
646 // Use the system SSL library (Secure Transport on Mac, SChannel on Windows)
647 // instead of NSS for SSL.
648 const char kUseSystemSSL[]                  = "use-system-ssl";
649 #endif
650
651 // Enable per-tile page painting.
652 const char kEnablePerTilePainting[]         = "enable-per-tile-painting";
653
654 // Disables the use of a 3D software rasterizer.
655 const char kDisableSoftwareRasterizer[]     = "disable-software-rasterizer";
656
657 #if defined(USE_AURA)
658 // Configures the time after a GestureFlingCancel in which taps are cancelled.
659 extern const char kFlingTapSuppressMaxDown[] = "fling-tap-suppress-max-down";
660
661 // Maximum time between mousedown and mouseup to be considered a tap.
662 extern const char kFlingTapSuppressMaxGap[] = "fling-tap-suppress-max-gap";
663
664 // Forces usage of the test compositor. Needed to run ui tests on bots.
665 extern const char kTestCompositor[]         = "test-compositor";
666 #endif
667
668 // Sets the tile size used by composited layers.
669 const char kDefaultTileWidth[]              = "default-tile-width";
670 const char kDefaultTileHeight[]             = "default-tile-height";
671
672 // Sets the width and height above which a composited layer will get tiled.
673 const char kMaxUntiledLayerWidth[]          = "max-untiled-layer-width";
674 const char kMaxUntiledLayerHeight[]         = "max-untiled-layer-height";
675
676 const char kFixedPositionCreatesStackingContext[]
677     = "fixed-position-creates-stacking-context";
678 }  // namespace switches
```