SOSCON

오픈 소스 패치를 통한 Chromium-blink 이해와 Contribution

오픈 소스 시작하기

삼성전자 | Think Tank Team | 김현준
삼성전자 | Common Platform Lab | 박중헌

발표내용

Chromium 소스 Download 및 Build

Download

#1 depot_tools 설치

```
$ git clone https://chromium.googlesource.com/chromium/tools/depot_tools.git
$ export PATH="$PATH:/path/to/depot_tools"
```

#2 Check out the code and its dependencies

```
$ mkdir ~/chromium && cd ~/chromium
$ fetch --nohooks chromium
```

#3 Additional build dependencies 설치

```
$ cd src
$ ./build/install-build-deps.sh
$ gclient runhooks
```

Chromium 소스 Download 및 Build

Build

#1 GN을 이용하여 Ninja file 생성

```
$ gn gen out/Default
```

#2 Chromium 빌드

```
$ autoninja -C out/Default chrome
```

#2 Content Shell 빌드

```
$ autoninja -C out/Default content_shell
```

Chromium 소스 Download 및 Build

Build 과정 설명

#1 GN File들을 통해 Ninja 파일 생성
(해당 과정에서 IDL파일을 읽어 Javascript binding 코드 및 CSS 관련 코드 등을 생성 함)

#2 Ninja 파일을 이용하여 컴파일

Chromium 소스 Download 및 Build

Javscript Binding 코드
IDL (interface description language)

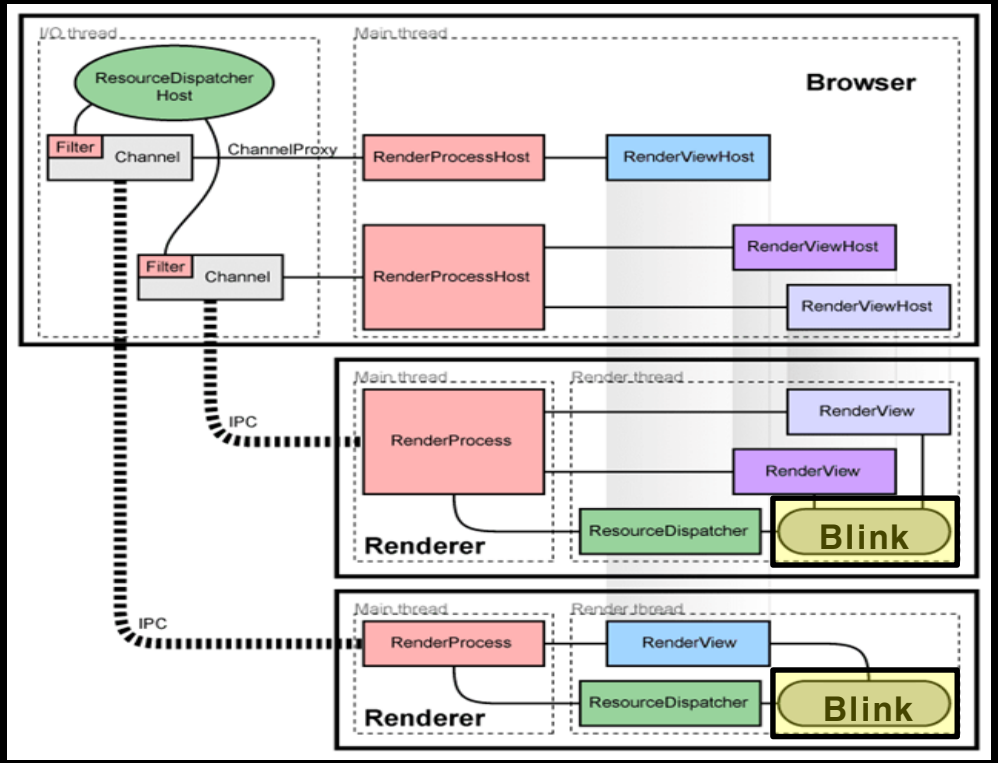
```
Attr? getAttributeNode(DOMString name);
Attr? getAttributeNodeNS(DOMString? namespaceURI, DOMString localName);
[RaisesException, CEReactions, CustomElementCallbacks] Attr? setAttributeNode(Attr attr);
[RaisesException, CEReactions, CustomElementCallbacks] Attr? setAttributeNodeNS(Attr attr);
[RaisesException, CEReactions, CustomElementCallbacks] Attr removeAttributeNode(Attr attr);
```

V8

Native code

```
Attr* Element::getAttributeNode(const AtomicString& local_name) {
  if (!GetElementData())
    return nullptr;
  SynchronizeAttribute(local_name);
  const Attribute* attribute =
      GetElementData()->Attributes().Find(LowercaseIfNecessary(local_name));
  if (!attribute)
    return nullptr;
  return EnsureAttr(attribute->GetName());
}
```

Chromium – Blink 동작 구조

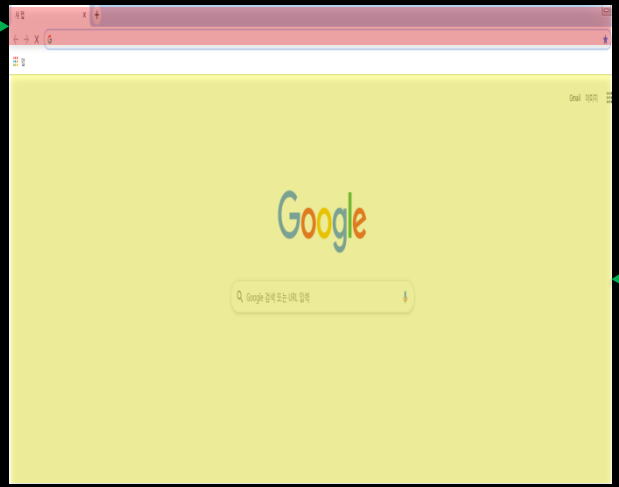


Chromium – Blink 동작 구조

Linux 에서 'ps'로 찍어보면

```
18810 18481 15 18:29 pts/1    00:00:00 /home/ttt/chromium/src/out/Default/content_shell --no-sandbox
18812 18810  2 18:29 pts/1    00:00:00 /home/ttt/chromium/src/out/Default/content_shell --type=zygote --no-sandbox
18829 18812  2 18:29 pts/1    00:00:00 /home/ttt/chromium/src/out/Default/content_shell --type=renderer --no-sandbox --field-tri
m-raster-threads=4 --enable-main-frame-before-activation --service-request-channel-token=4784992174345180916 --renderer-client-i
```

Browser Process → Fork → Zygote Process → Fork → Renderer Process



이슈 카테고리 및 카테고리 설명

Chromium Mailing List

[blink-dev] Blink bug status as of 2019-09-23 받은편지함 ×

TAMURA, Kent <tkent@chromium.org>
blink-dev에게 수신거부

영어 > 한국어 메일 번역

Blink bug status as of 2019-09-23

Component	Open	Slow Triage	Pri-0/1	No owner	Oldest
Whole Blink	13,701 (+65)	704 (+17)	881 (-10)	158	Apr 2014
Uncategorized	24 (-2)	0	1	1	83 minutes ago
Blink>Accessibility	179 (+3)	18 (-3)	24 (+2)	8	Sep 2016
Blink>Animation	154 (+10)	1	3	0	Jun 21
Blink>AppManifest	13	5	0	0	
Blink>BackgroundFetch	10 (+1)	0	2 (+1)	1	Sep 2018
Blink>BackgroundSync	26 (-2)	0	2	0	Aug 2018
Blink>Bindings	259 (+6)	5	11 (+1)	2	Sep 2015
Blink>Bluetooth	124	8	0	0	
Blink>Canvas	276 (+6)	0 (-1)	13 (-1)	0	May 2017
Blink>Compositing	226 (+3)	0	10 (+1)	3	Oct 2017
Blink>Contacts	11 (-1)	0	2 (-1)	0	Jun 27
Blink>CSS	431 (+4)	2 (-1)	4	1	May 18
Blink>DarkMode	65 (-2)	12 (-2)	6 (-5)	1	May 16
Blink>DataTransfer	176 (+2)	3 (+2)	9	3	Nov 2017
Blink>DOM	146 (-5)	3 (+1)	22 (+1)	11	Jun 2018
Blink>Editing	860 (-8)	4 (+1)	7 (-2)	3	Aug 16
Blink>FeaturePolicy	49 (+1)	0	1 (+1)	1	3 days ago
Blink>Fonts	301 (+2)	0 (-4)	13 (+1)	1	Nov 2015
Blink>Forms	362 (-4)	7 (+2)	10 (-2)	6	Sep 2018
Blink>Fullscreen	31 (-1)	1 (-2)	0	0	
Blink>GamepadAPI	72 (-3)	0	2 (-2)	0	Feb 2019

https://bugs.chromium.org

bugs.chromium.org/p/chromium/issues/list?q=Component%3ABlink>CSS&can=2

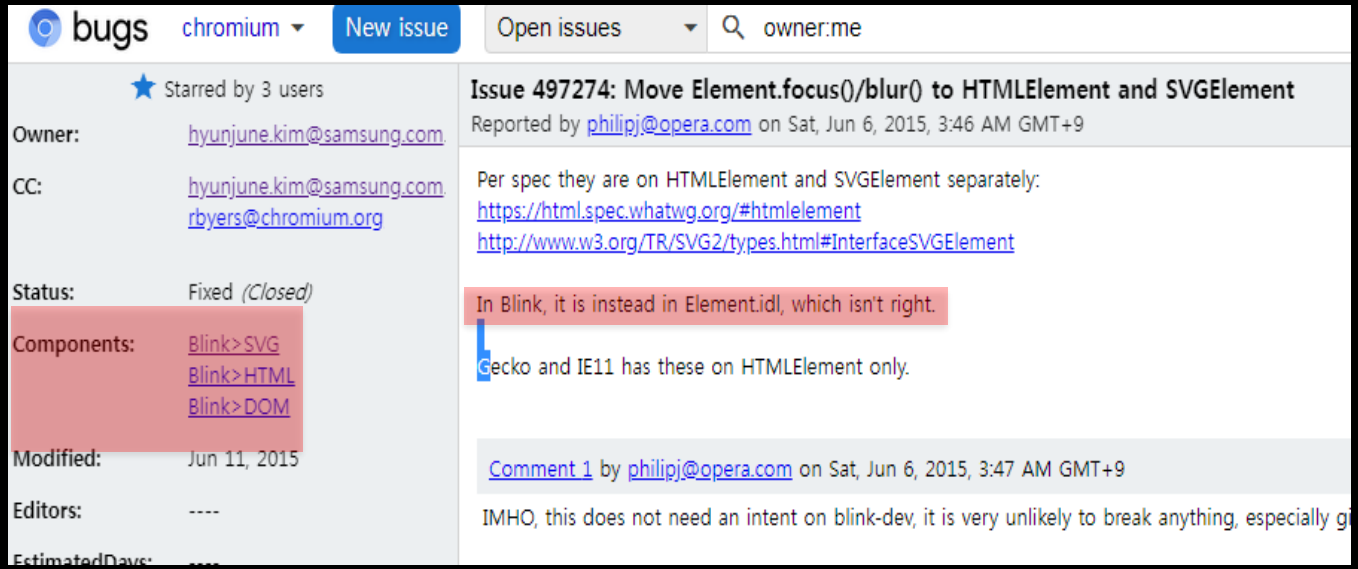
bugs chromium New issue Open issues Component:Blink>CSS

Select: All None Actions...

ID	Pri	M	Stars	ReleaseBlock	Component	Status	Owner	Summary + Labels
1008471	3	----	1	----	Blink>CSS	Available	----	<number> calc() is fully resolved at parse time
1008146	3	----	1	----	Blink>CSS	Available	----	Summation should sort terms in the specified order
1007985	3	----	1	----	Blink>CSS	Available	----	'translate' property does not obey serialization rules for getCompute
1007978	3	----	2	----	Blink>CSS	Available	----	'scale' property does not obey serialization rules for getComputedSt
1007624	----	----	1	----	Blink>CSS	Untriaged	----	[WPT] New failures introduced in external/wpt/css/css-backgrounds

이슈 카테고리 및 카테고리 설명

예시) 이슈 분석 :
Element내 focus와 blur라는 JS 함수가 있는데 표준에 따르면 HTMLElement와 SVGElement가 가지고 있어야 한다.
현재 Blink는 Element가 해당 함수를 가지고 있어서 잘 못 됐다.



이슈 카테고리 및 카테고리 설명

이슈 해결 : 이런 식으로 요구한 내용대로 수정하여 패치를 만들면 된다 (해당 이슈는 가장 난이도가 낮은 수준 임)

```
--- a/Source/core/dom/Element.idl
+++ b/Source/core/dom/Element.idl
@@ -111,10 +111,8 @@

     // HTML
     // https://html.spec.whatwg.org/#htmlelement
-    // FIXME: dataset, focus() and blur() should be on HTMLElement. crbug.com/497274
+    // FIXME: dataset should be on HTMLElement.
     [SameObject, PerWorldBindings] readonly attribute DOMStringMap dataset;
-    void focus();
-    void blur();

     // Non-standard APIs
     // https://www.w3.org/Bugs/Public/show_bug.cgi?id=19962
```

```
--- a/Source/core/html/HTMLElement.idl
+++ b/Source/core/html/HTMLElement.idl
@@ -32,7 +32,8 @@ interface HTMLElement : Element {
     [Reflect] attribute boolean hidden;
     void click();
     [CustomElementCallbacks] attribute long tabIndex;
-    // FIXME: focus() and blur() are on Element. crbug.com/497274
+    void focus();
+    void blur();
     [Reflect] attribute DOMString accessKey;
     [CustomElementCallbacks] attribute boolean draggable;
     [RuntimeEnabled=ContextMenu] attribute HTMLMenuElement? contextMenu;
```

```
--- a/Source/core/svg/SVGElement.idl
+++ b/Source/core/svg/SVGElement.idl
@@ -29,7 +29,8 @@ interface SVGElement : Element {
     readonly attribute SVGSVGElement? ownerSVGElement;
     readonly attribute SVGElement? viewportElement;
     [CustomElementCallbacks] attribute long tabIndex;
-    // TODO(philipj): focus() and blur() are on Element. crbug.com/497274
+    void focus();
+    void blur();
 };
```

Debug

#1 GDB를 활용

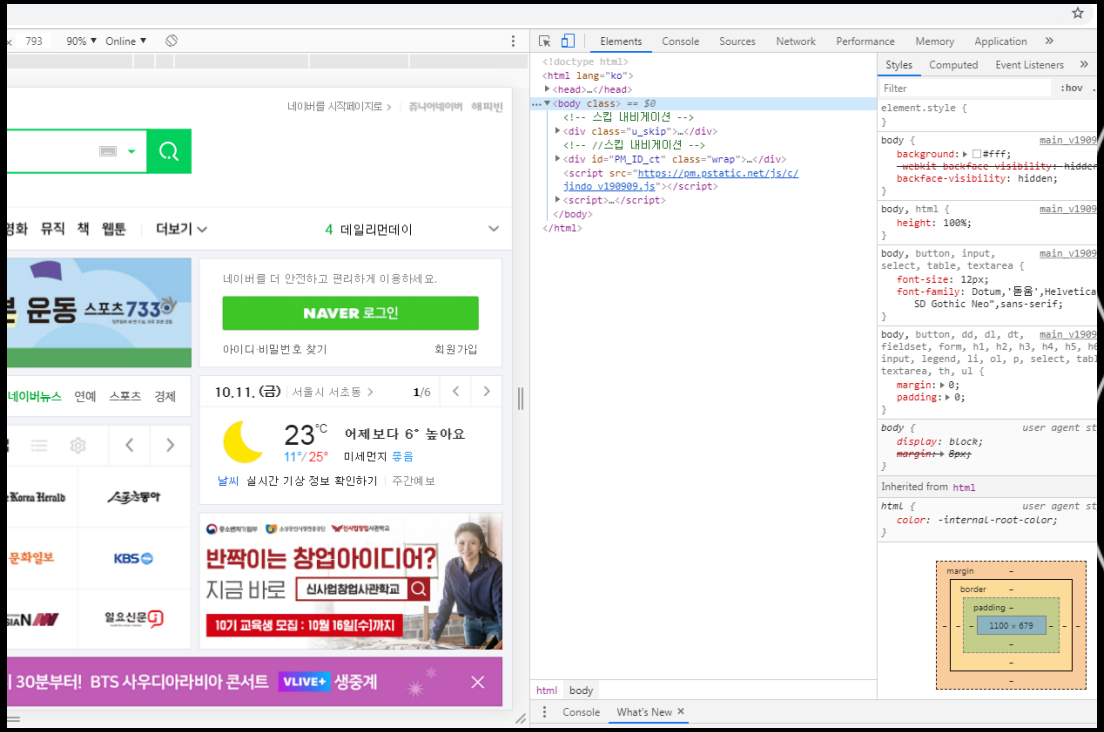
$ gdb –args ./out/mybuild/content_shell –no-sandbox

```
ttt@ttt-500T8A-500S8A:~/chromium/src$ gdb --args ./out/mybuild/content_shell --no-sandbox
GNU gdb (Ubuntu 8.1-0ubuntu3) 8.1.0.20180409-git
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./out/mybuild/content_shell...done.

warning: Could not find DWO CU obj/content/shell/content_shell/shell_main.dwo(0x21cc35e450197d1c) referenced b
(gdb) r
Starting program: /home/ttt/chromium/src/out/mybuild/content_shell --no-sandbox
```

#2 Flag를 통한 Layout tree dump
$./out/Default/content_shell –run-web-tests ./bug/test.html

```
ttt@ttt-500T8A-500S8A:~/chromium/src$ ./out/Default/content_shell  --run-web-tests ./bug/test.html
[20332:20332:1011/193234.305074:INFO:content_main_runner_impl.cc(943)] Chrome is running in full browser mode.

DevTools listening on ws://127.0.0.1:46491/devtools/browser/111bbde4-a7a1-43f8-96f8-233abcf19706
#READY
layer at (0,0) size 800x600 clip at (0,0) size 800x585 scrollWidth 821
  LayoutView at (0,0) size 800x600
layer at (0,0) size 800x585
  LayoutNGBlockFlow {HTML} at (0,0) size 800x585
    LayoutNGBlockFlow {BODY} at (8,8) size 784x569
      LayoutText {#text} at (0,0) size 22x19
        text run at (0,0) width 22: "abc"
      LayoutInline {SPAN} at (806,0) size 7x19
        LayoutText {#text} at (806,0) size 7x19
          text run at (806,0) width 7: "a"
      LayoutText {#text} at (0,0) size 0x0
[20332:20341:1011/193235.706160:WARNING:discardable_shared_memory_manager.cc(432)] Some MojoDiscardableSharedMemo
[20332:20406:1011/193235.710707:WARNING:internal_linux.cc(64)] Failed to read /proc/20351/stat
```

#3 Inspector


Elements
Console
Sources
Network
Performance
Memory
Application
Styles
Computed
Event Listeners
네이버
NAVER 로그인
반짝이는 창업아이디어?
지금 바로
신사업창업사관학교
10기 교육생 모집 : 10월 16일(수)까지
margin
border
padding
1100 × 679

이슈 사례를 통한 Contribution

Issue None : Typo :)

```
@@                    +10↑ - Show 275 common lines - +10↓
276     return;
277   wtf_size_t end_offset = text.find(kNewlineCharacter, offset_);
278   base_direction_ = NGBidiParagraph::BaseDirectionForString(
279       end_offset == kNotFound
280           ? StringView(text, offset_)
281           : StringView(text, offset_, end_offset - offset_));
282 }
283
284 // Initialize internal states for the next line.
285 void NGLineBreaker::PrepareNextLine(NGLineInfo* line_info) {
286   // NGLineInfo is not supposed to be re-used becase it's not much gain and to
287   // avoid rare code path.
288   const NGInlineItemResults& item_results = line_info->Results();
289   DCHECK(item_results.IsEmpty());
290
291   if (item_index_) {
292     // We're past the first line
293     previous_line_had_forced_break_ = is_after_forced_break_;
294     is_after_forced_break_ = false;
295     is_first_formatted_line_ = false;
296     use_first_line_style_ = false;
@@                    +10↑ - Show 1614 common lines - +10↓
```

```
@@                    +10↑ - Show 275 common lines - +10↓
276     return;
277   wtf_size_t end_offset = text.find(kNewlineCharacter, offset_);
278   base_direction_ = NGBidiParagraph::BaseDirectionForString(
279       end_offset == kNotFound
280           ? StringView(text, offset_)
281           : StringView(text, offset_, end_offset - offset_));
282 }
283
284 // Initialize internal states for the next line.
285 void NGLineBreaker::PrepareNextLine(NGLineInfo* line_info) {
286   // NGLineInfo is not supposed to be re-used because it's not much gain and to
287   // avoid rare code path.
288   const NGInlineItemResults& item_results = line_info->Results();
289   DCHECK(item_results.IsEmpty());
290
291   if (item_index_) {
292     // We're past the first line
293     previous_line_had_forced_break_ = is_after_forced_break_;
294     is_after_forced_break_ = false;
295     is_first_formatted_line_ = false;
296     use_first_line_style_ = false;
@@                    +10↑ - Show 1614 common lines - +10↓
```

이슈 사례를 통한 Contribution

Issue 463435 : Iframe.sandbox must have readonly on idl file.

What steps will reproduce the problem?

When run this script
<iframe id="if7"/>
<script>
var if_temp = if7;
if_temp.sandbox = "not readonly";
alert(if_temp.sandbox);
</script>,
modify iframe.sandbox value.

What is the expected result?
Not change 'iframe.sandbox' value.

#http://dev.w3.org/html5/spec-preview/the-iframe-element.html

I want that change blink's idl file named 'HTMLIFrameElement.idl' to insert 'readonly' for sandbox.

Thank you.

이슈 사례를 통한 Contribution

Issue 463435 : Iframe.sandbox must have readonly on idl file.

문제 : Iframe element 내 sandbox 속성이 readonly가 아님
'표준'과 다르게 구현 됨
'sandbox'의 데이터 타입이 DOMString에서 DOMSettableToken으로 변경 됨
해결 : 단순히 Idl 만 바꾸는 문제가 아닌 '디자인 패턴'을 적용

```
> @fs
> How about this. Could i move m_sandbox to HTMLFrameOwnerElement?
> like this one.
> And when call setSandboxFlags,
>  void HTMLFrameOwnerElement::setSandboxFlags(SandboxFlags flags)
> {
>     m_sandbox = UpdateSandbox(flags);
>     m_sandboxFlags = flags;
>     // Don't notify about updates if contentFrame() is null, for example when
>     // the subframe hasn't been created yet.
>     if (contentFrame())
>
> document().frame()->loader().client()->didChangeSandboxFlags(contentFrame(),
> flags);
> }
>
> When call sandboxFlags
> SandboxFlags HTMLFrameOwnerElement::sandboxFlags() const
> {
>     String invalidTokens;
>     if (!m_sandbox->length())
>         return SandboxNone;
>     return parseSandboxPolicy(m_sandbox->value(), invalidTokens);
> }
>
> then SandboxFlags Value is kept up-to-date.

Well, one way would be to add change-notifications on DOMSettableTokenList.
```

```
-DOMSettableTokenList::DOMSettableTokenList()
+DOMSettableTokenList::DOMSettableTokenList(DOMSettableTokenListObserver* observer)
     : m_value()
     , m_tokens()
+    , m_observer(observer)
 {
 }

@@ -89,6 +90,8 @@ void DOMSettableTokenList::setValue(const AtomicString& value)
 {
     m_value = value;
     m_tokens.set(value, false);
+    if (m_observer)
+        m_observer->valueChanged();
 }
```

이슈 사례를 통한 Contribution

Issue 477545 : SVG text selection has bugs with collapsed whitespace

Consider the following (http://jsfiddle.net/progers/g1z286fL/):

```
<svg width="400" height="200">
    <text x="20" y="20">Text selection does not work with collapsed whitespace.</text>
    <text x="20" y="40">Select from world to hello below:</text>
    <text x="20" y="80"> hello world </text>
    <text x="20" y="120">      hello          world       </text>
</svg>
```

I think we just need to switch a textLength() call to a renderedTextLength() call somewhere.

문제 : 아래 SVG 문서를 동작시켰을 때 world부터 hello까지
마우스로 끌어서 선택(text selection)하면 잘 못 선택 됨

이슈 사례를 통한 Contribution

Issue 477545 : SVG text selection has bugs with collapsed whitespace

해결 : SVG의 Text selection 코드만 몇 일 본 결과 위치를 발견했고 Log 및 GDB를 통해 문제를 파악 함

문제는 마우스를 기준으로 가장 가까운 fragmentRect(chunk)를 잘못 찾아서 생긴 이슈

```
+static float squaredDistanceToClosestPoint(const FloatRect& rect, const FloatPoint& point)
+{
+    FloatPoint closestPoint;
+    closestPoint.setX(std::max(std::min(point.x(), rect.maxX()), rect.x()));
+    closestPoint.setY(std::max(std::min(point.y(), rect.maxY()), rect.y()));
+    return (point - closestPoint).diagonalLengthSquared();
+}
+
```

```
-    float baseline = m_scaledFont.fontMetrics().floatAscent();
+    ASSERT(m_scalingFactor);
+    float baseline = m_scaledFont.fontMetrics().floatAscent() / m_scalingFactor;

     LayoutBlock* containingBlock = this->containingBlock();
     ASSERT(containingBlock);
@@ -182,12 +191,14 @@ PositionWithAffinity LayoutSVGInlineText::positionForPoint(const LayoutPoint& po
             const SVGTextFragment& fragment = fragments.at(i);
             FloatRect fragmentRect(fragment.x, fragment.y - baseline, fragment.width, fragment.height);
             fragment.buildFragmentTransform(fragmentTransform);
-            fragmentRect = fragmentTransform.mapRect(fragmentRect);
+            if (!fragmentTransform.isIdentity())
+                fragmentRect = fragmentTransform.mapRect(fragmentRect);

-            float distance = powf(fragmentRect.x() - absolutePoint.x(), 2) +
-                powf(fragmentRect.y() + fragmentRect.height() / 2 - absolutePoint.y(), 2);
+            float distance = 0;
+            if (!fragmentRect.contains(absolutePoint))
+                distance = squaredDistanceToClosestPoint(fragmentRect, absolutePoint);

-            if (distance < closestDistance) {
+            if (distance <= closestDistance) {
                 closestDistance = distance;
                 closestDistanceBox = textBox;
```

이슈 사례를 통한 Contribution

Issue 571723: When has currentColor, … it should be inherited as currentColor

문제 : 'currentColor'가 동적으로 변경되지 않는 상황

해결 : SVG에서'currentColor'를 가지고 있을 때 Difference 체크에 대한 고려가 안 돼있어서 **Style이 반영이 안 된 경우**

What steps will reproduce the problem?

1. run this example.

```
<svg xmlns="http://www.w3.org/2000/svg">
  <g fill="currentColor" color="red">
    <rect width="100" height="100" color="green"/>
  </g>
</svg>
```

2. Currently draw the red rectangle,
3.

What is the expected output?

Expected result :

The green rectangle.

```
@@ -1796,7 +1796,9 @@ StyleDifference LayoutObject::adjustStyleDifference(StyleDifference diff) const
             // skipped or we will miss invalidating decorations (e.g., underlines).
             || (isText() && !isBR() && toLayoutText(this)->hasTextBoxes())
             // Caret is painted in text color.
-            || (isLayoutBlock() && toLayoutBlock(this)->hasCaret()))
+            || (isLayoutBlock() && toLayoutBlock(this)->hasCaret())
+            || (isSVG() && style()->svgStyle().isFillColorCurrentColor())
+            || (isSVG() && style()->svgStyle().isStrokeColorCurrentColor()))
             diff.setNeedsPaintInvalidationObject();
     }
```

이슈 사례를 통한 Contribution

Issue 543061 : document.title in an SVG document should only return titles that are children of the root

문제 : SVG 문서에 대해서 document.title(Javascirpt) property가 표준과 다르게 구현된 경우
해결 : 표준 문서를 보고 써있는 그대로 구현
표준문서 : https://html.spec.whatwg.org/multipage/dom.html#document.title

↳ **If the document element is an SVG svg element**

1. If there is an SVG title element that is a child of the document element, let *element* be the first such element.
2. Otherwise:
    1. Let *element* be the result of creating an element given the document element's node document, title, and
    2. Insert *element* as the first child of the document element.
3. String replace all with the given value within *element*.

```
void Document::setTitleElement(Element* titleElement)
{
-    // Only allow the first title element to change the title -- others have no effect.
-    if (m_titleElement && m_titleElement != titleElement) {
-        if (isHTMLDocument() || isXHTMLDocument()) {
+    // If the root element is an svg element in the SVG namespace, then let value be the child text content
+    // of the first title element in the SVG namespace that is a child of the root element.
+    if (isSVGSVGElement(documentElement())) {
+        m_titleElement = Traversal<SVGTitleElement>::firstChild(*documentElement());
+    } else {
+        if (m_titleElement && m_titleElement != titleElement)
+            m_titleElement = Traversal<HTMLTitleElement>::firstWithin(*this);
-        } else if (isSVGDocument()) {
-            m_titleElement = Traversal<SVGTitleElement>::firstWithin(*this);
+        else
+            m_titleElement = titleElement;
+
+        // If the root element isn't an svg element in the SVG namespace and the title element is
+        // in the SVG namespace, it is ignored.
+        if (isSVGTitleElement(m_titleElement)) {
+            m_titleElement = nullptr;
+            return;
+        }
-    } else {
-        m_titleElement = titleElement;
    }
```

오픈소스 패치를 통한 CSS, Layout에 대한 이해

그리고 Blink, WebKit에 Contribution하는 방법

발표내용

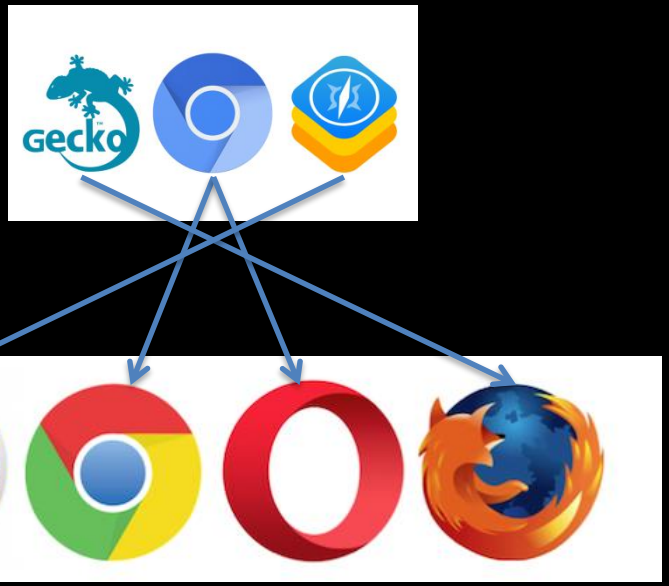
For whom? For What?

세미나 대상
- CSS 및 Layout 코드 hack의 entry point를 찾는 사람

세미나를 통해 기대하는 것
- Css 및 layout의 코드를 보는 계기
- 패치 커밋 과정에서 Web Engine, csswg community와 소통하고 소소하게 Spec을 변경하는 과정에 기여하는 활동 살펴보기

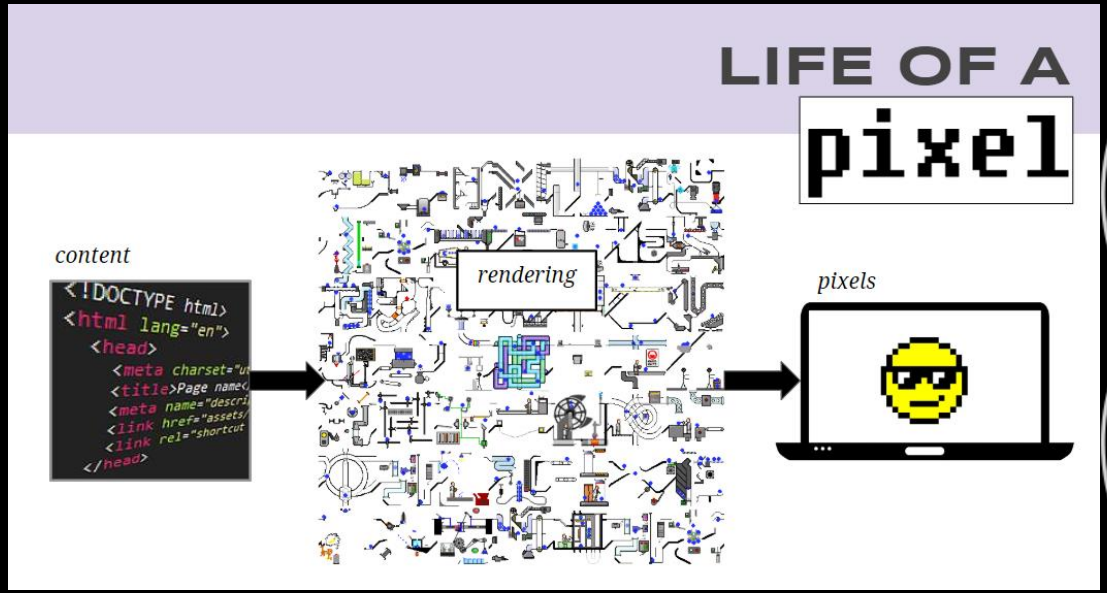
브라우저와 웹엔진

Safari – WebKit
Chrome, Opera – Chromium(Blink)
FireFox - Gecko



웹엔진이란? (1)

HTML, CSS, Javascript 등의 input을 받아서
그 결과물을 화면에 표시해주는 브라우저의 핵심 요소.

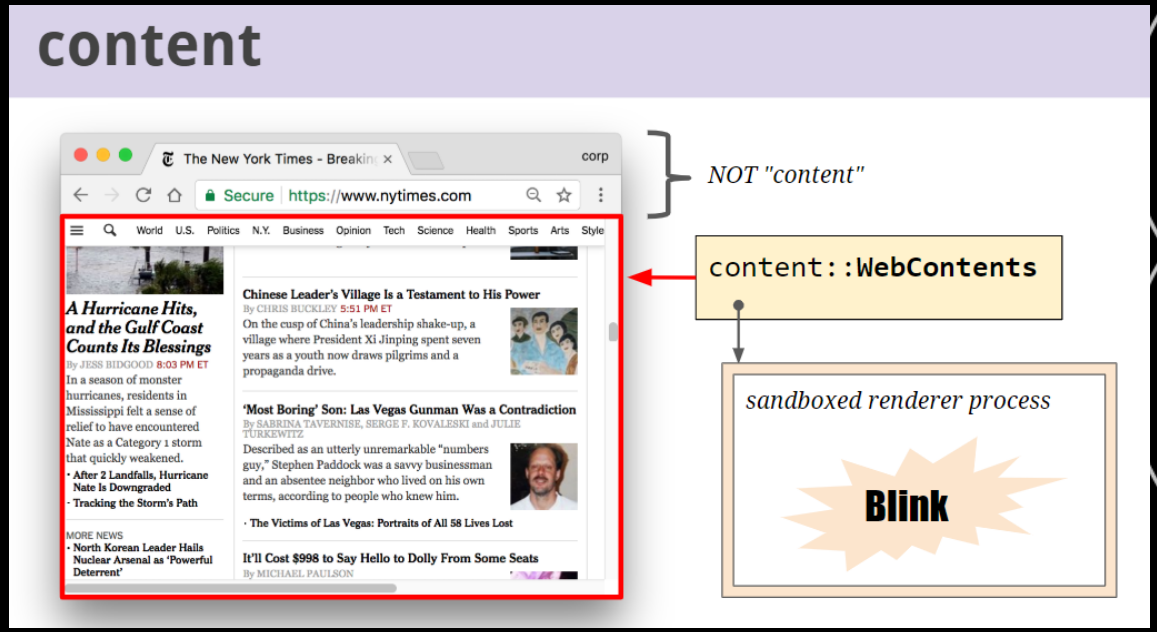


http://bit.ly/lifeofapixel

웹엔진이란? (2)

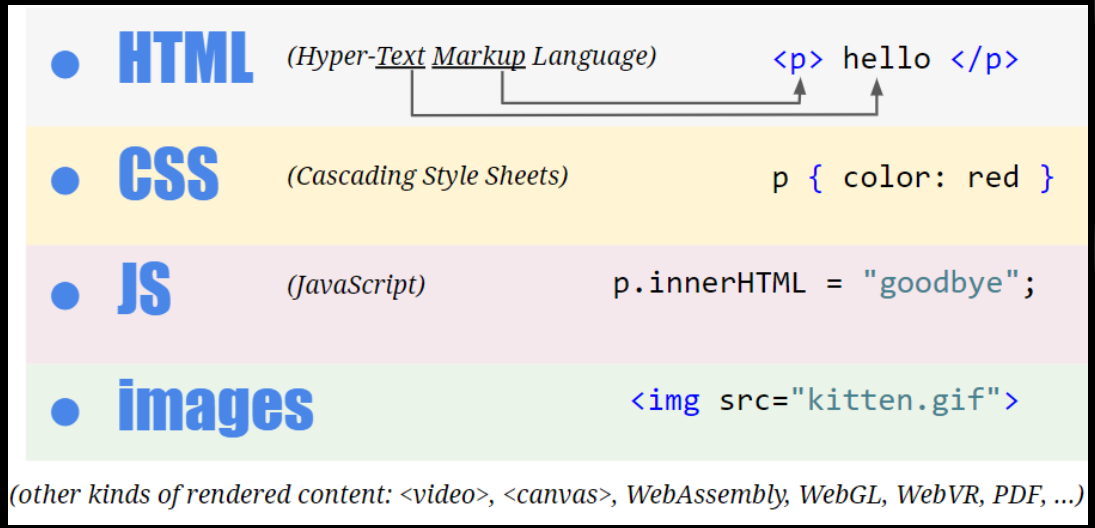
Red Box 안의 웹페이지 - Chromium의 web engine인 Blink가 그린 결과물

- Chromium Content Module - Multi process로 이루어진 Chromium의 구조를 담고 있으며, 빨간박스 내의 모든 것을 담당한다.



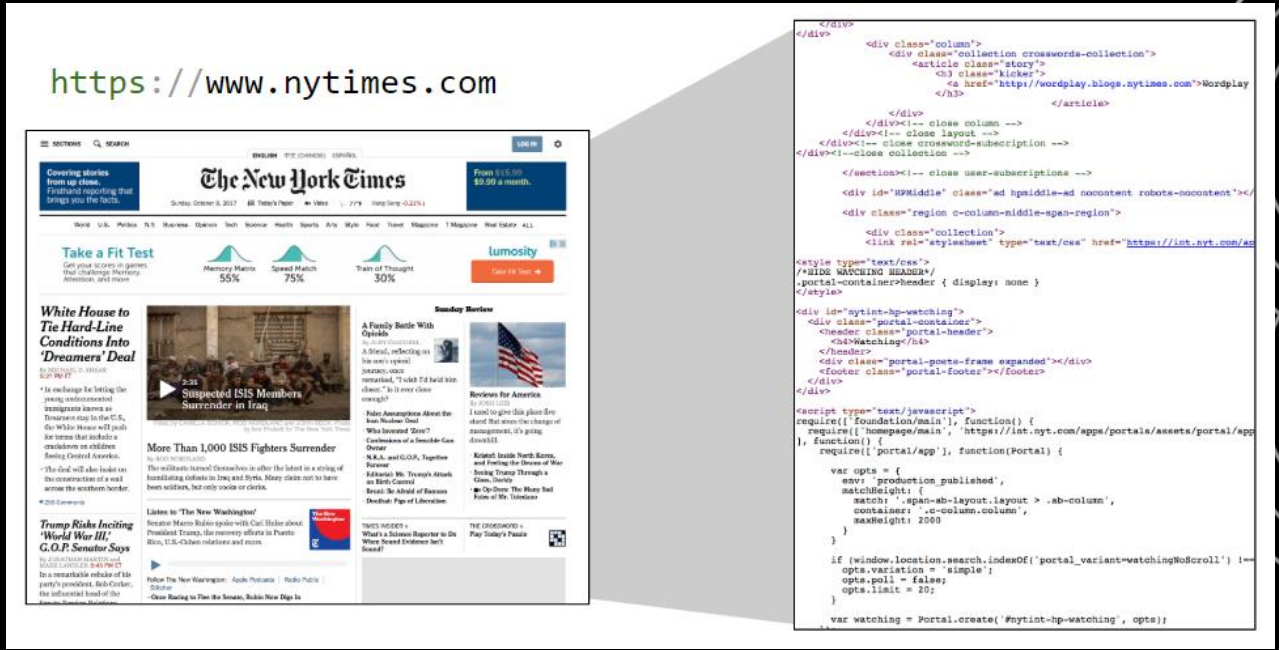
Web Page Content

- HTML – markup 언어. 문서를 구조화시켜주는 의미요소들을 포함, e.g. <p>, ,...
- CSS – markup element가 어떻게 그려질지 지정.
- JS – 위 요소들을 동적으로 변경 가능.



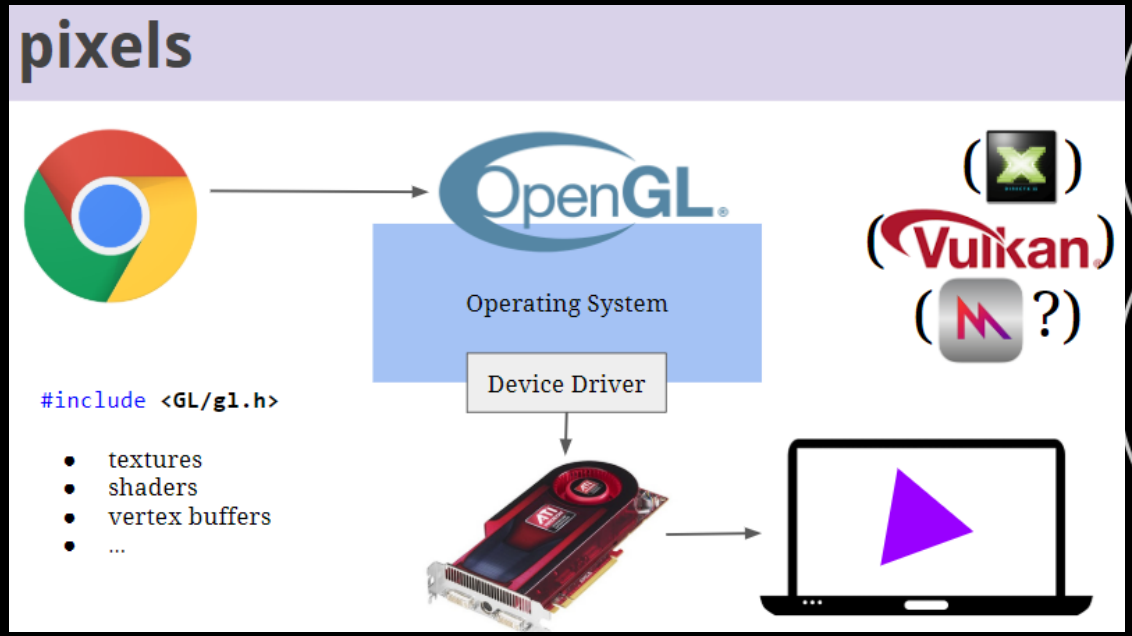
A Real Web Page Example

- 실제 웹페이지는 plain text로서 네트워크를 통해 전달되는 수천줄의 HTML, CSS, Javascript로 이루어진다.
- 컴파일이나 패키징 등의 개념없이 이들 리소스가 직접 Web Engine의 input이 된다.



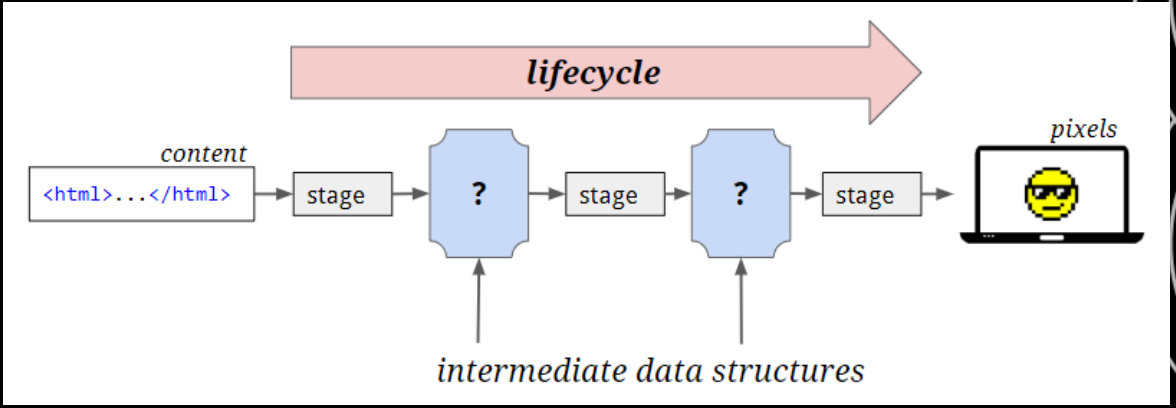
Pixels

- Input resource들이 rendering pipeline을 거쳐서 최종적으로 pixel들을 screen에 출력
- Screen에 pixel을 출력하기 위해 OS에서 제공하는 graphics library를 사용
- OS graphics library를 사용하기 위해 오늘날 대부분의 platform에서 제공하는 표준 API를 사용, OpenGL(나중엔 Vulkan, Metal…)



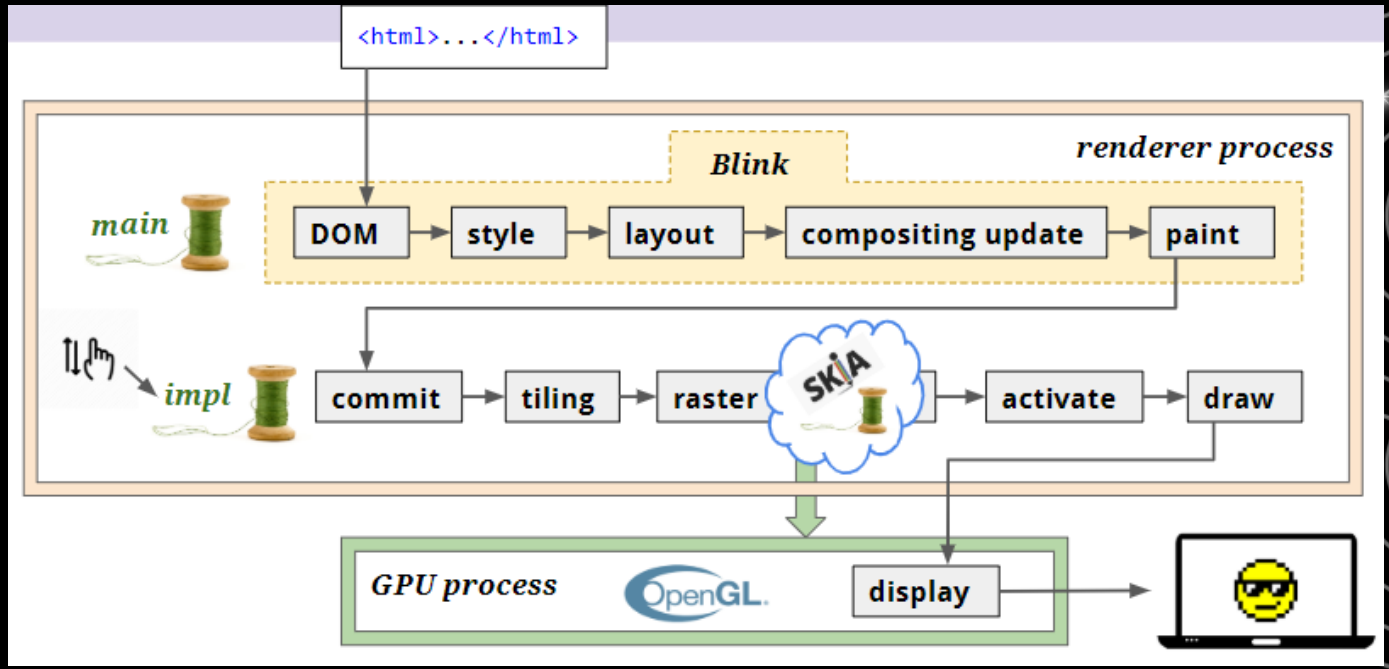
Stages

- Rendering - Input resource를 최종적으로 스크린위의 픽셀로 표현하기까지의 과정
- Rendering pipeline은 여러 개의 lifecycle stage로 쪼갤 수 있다.
- 각 단계마다 중간 결과물이 출력되고 이는 곧 다음단계의 입력이 된다.



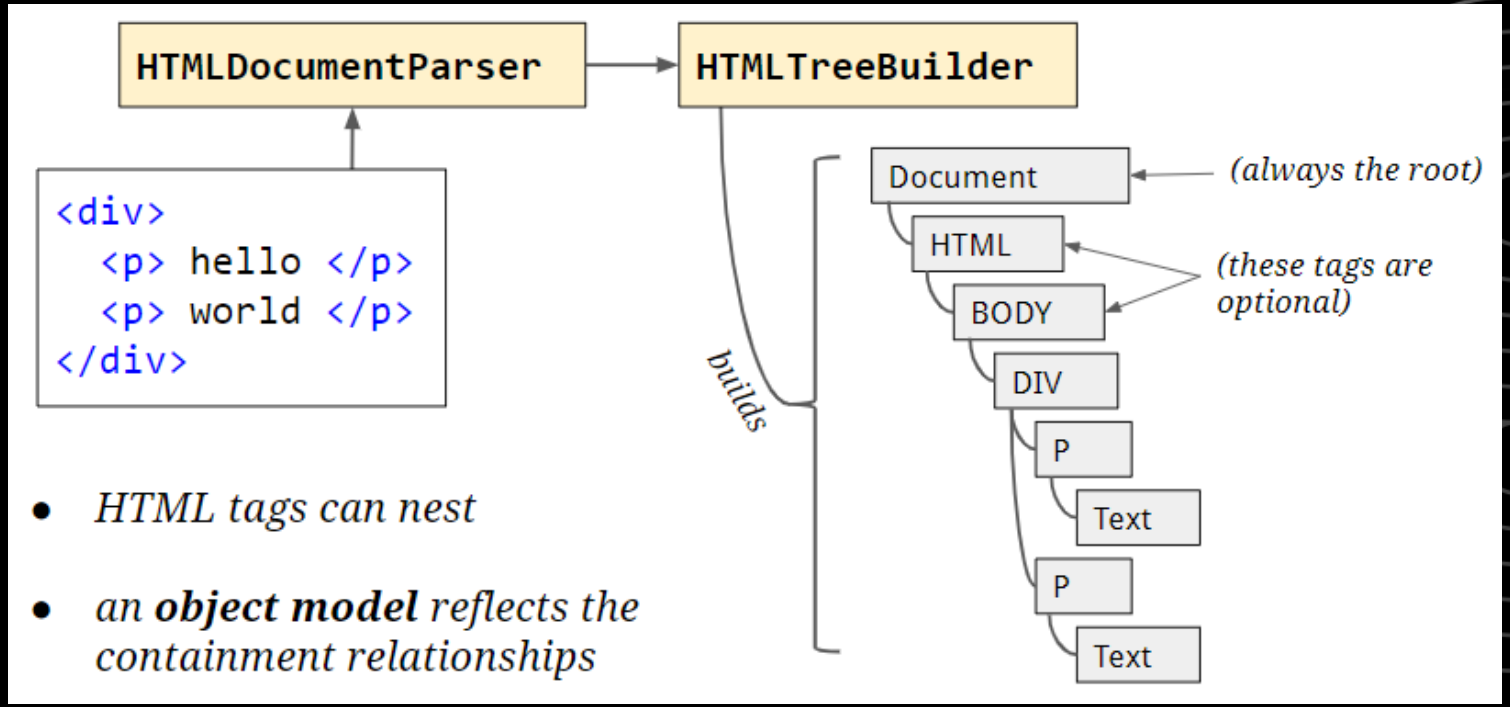
The Entire Rendering Pipeline

- Input resource를 받아서 pixel을 스크린에 출력하기까지의 전체 과정



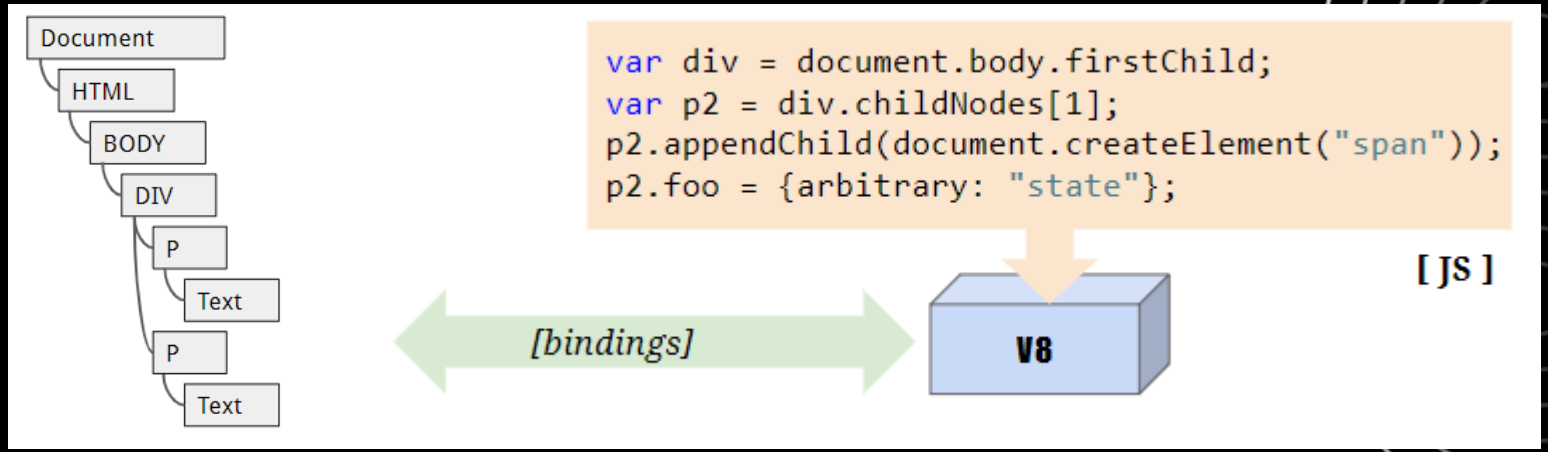
Parsing

- HTML을 읽어들인 후 elements hierarchy를 고려해서 DOM(Document Object Model) tree로 변환한다.



DOM

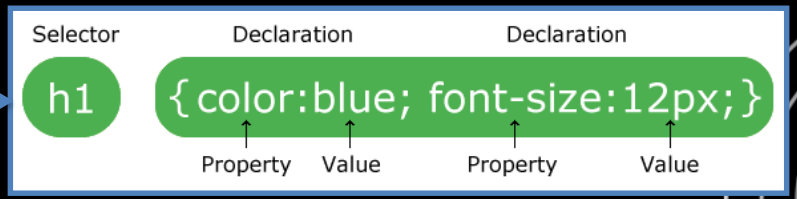
- Blink의 HTML을 표현하는 내부 표현방식
- Chromium의 Javascript Engine인 V8은 Binding이라고 불리는 system을 통해 실제 DOM tree와 소통하는 DOM Web API를 제공한다.

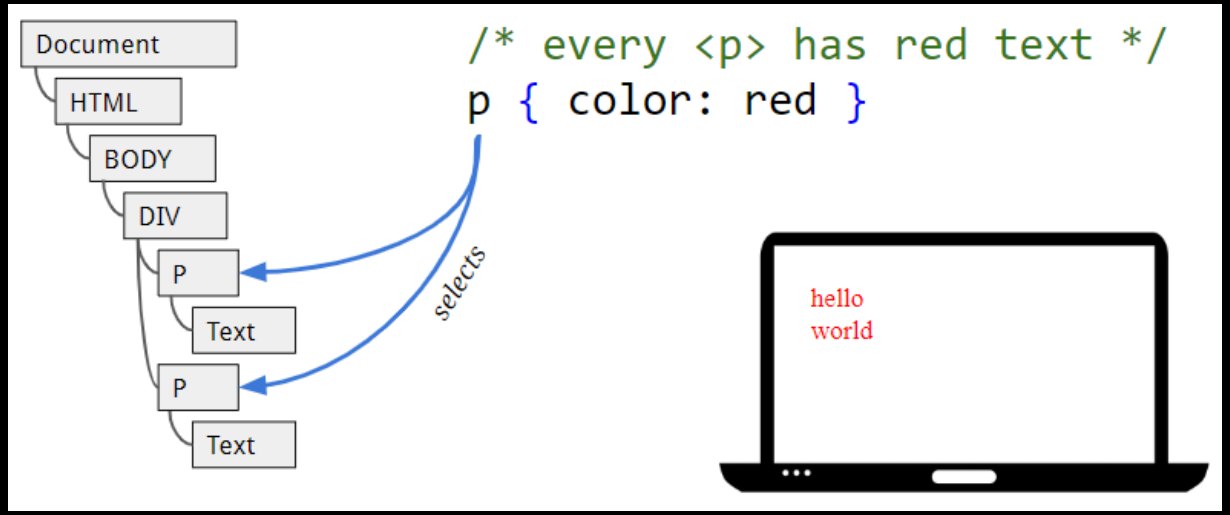


Style (1)

- DOM tree를 만든 후, 다음 단계로서 CSS Style을 처리한다.
- CSS Selector는 CSS property declaration이 적용될 DOM tree의 subset을 선택한다.

A CSS Ruleset
(In short, CSS Rule)





https://www.w3schools.com/css/css_syntax.asp

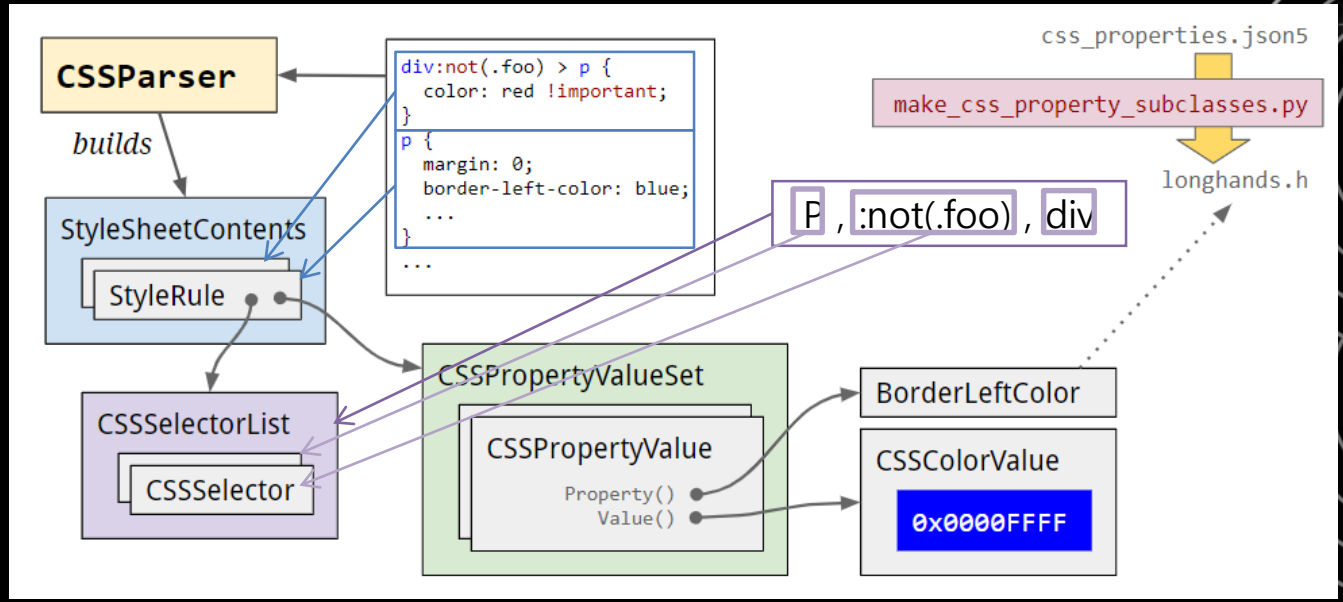
Style (2)

- CSS: Web Author들이 DOM Element가 어떻게 그려질지 지정하는 도구
- 수백가지의 css property들이 존재

`font-weight: bold;`	**hello**
`margin-left: 2cm;`	hello
`outline: dashed blue;`	hello
`transform: rotate(20deg);`	*hello*
`background: url(kitten.jpg);`	hello 

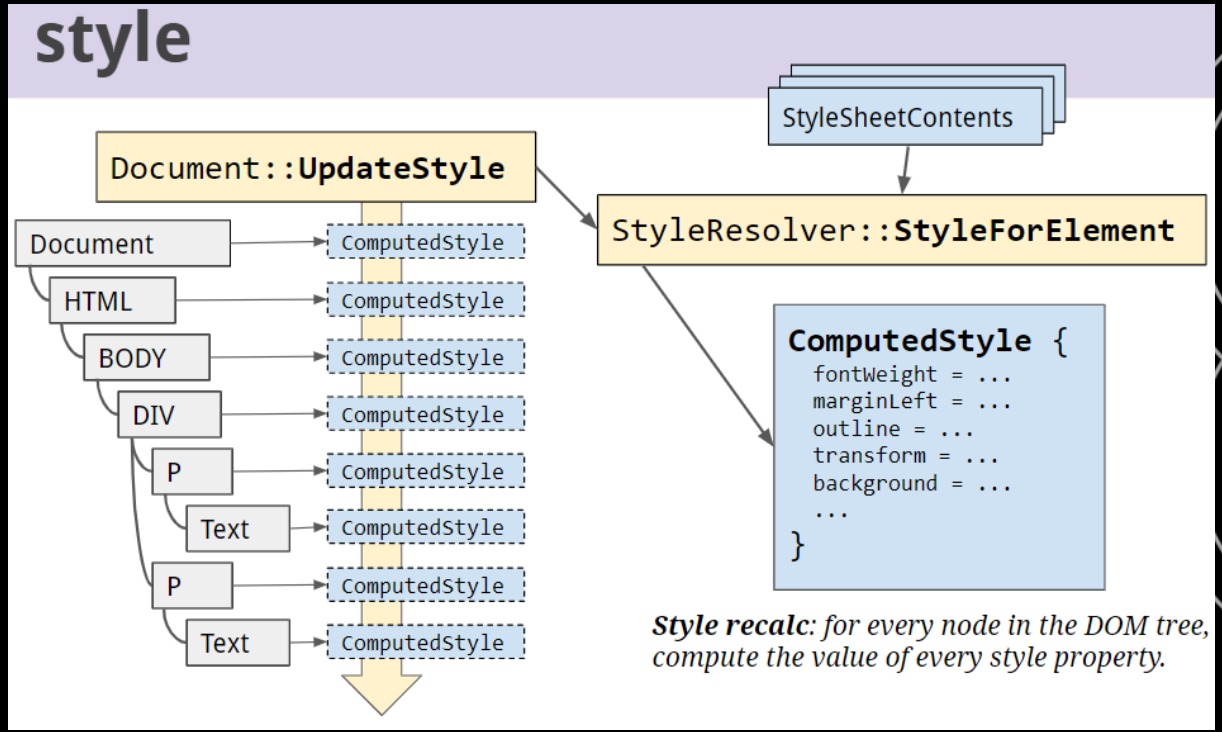
Style (3)

- CSSParser는 기술된 css style을 parsing해서 CSS Specification을 반영하는 Style Rule에 대한 객체 모델을 만들어낸다.



Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.



Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
void Document::UpdateStyle() {
  DCHECK(!View()->ShouldThrottleRendering());
  TRACE_EVENT_BEGIN0("blink,blink_style", "Document::updateStyle");
  RUNTIME_CALL_TIMER_SCOPE(V8PerIsolateData::MainThreadIsolate(),
                           RuntimeCallStats::CounterId::kUpdateStyle);

  unsigned initial_element_count = GetStyleEngine().StyleForElementCount();

  lifecycle_.AdvanceTo(DocumentLifecycle::kInStyleRecalc);

  StyleResolver& resolver = EnsureStyleResolver();

  bool should_record_stats;                (.....)
  TRACE_EVENT_CATEGORY_GROUP_ENABLED("blink,blink_style", &should_record_stats);
  GetStyleEngine().SetStatsEnabled(should_record_stats);

  if (Element* document_element = documentElement()) {
    if (change.TraverseChild(*document_element)) {
      TRACE_EVENT0("blink,blink_style", "Document::recalcStyle");
      SCOPED_BLINK_UMA_HISTOGRAM_TIMER_HIGHRES("Style.RecalcTime");
      Element* viewport_defining = ViewportDefiningElement();
      GetStyleEngine().RecalcStyle(change);
      if (viewport_defining != ViewportDefiningElement())
        ViewportDefiningElementDidChange();
    }
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
void StyleEngine::RecalcStyle(const StyleRecalcChange change) {
  DCHECK(GetDocument().documentElement());
  DCHECK(GetDocument().ChildNeedsStyleRecalc() || change.RecalcDescendants());

  Element& root_element = style_recalc_root_.RootElement();
  if (change.RecalcChildren() ||
      &root_element == GetDocument().documentElement()) {
    GetDocument().documentElement()->RecalcStyle(change);
  } else {
    Element* parent = root_element.ParentOrShadowHostElement();
    DCHECK(parent);
    SelectorFilterAncestorScope filter_scope(*parent);
    root_element.RecalcStyle(change);
  }
  for (ContainerNode* ancestor = root_element.ParentOrShadowHostNode();
       ancestor; ancestor = ancestor->ParentOrShadowHostNode()) {
    if (ancestor->IsElementNode())
      ToElement(ancestor)->RecalcStyleForTraversalRootAncestor();
    ancestor->ClearChildNeedsStyleRecalc();
  }
  style_recalc_root_.Clear();
}
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
void Element::RecalcStyle(const StyleRecalcChange change) {
  DCHECK(InActiveDocument());
  DCHECK(GetDocument().InStyleRecalc());
  DCHECK(!GetDocument().Lifecycle().InDetach());

  if (StyleRecalcBlockedByDisplayLock())
    return;

  if (HasCustomStyleCallbacks())
    WillRecalcStyle(change);

  StyleRecalcChange child_change = change.ForChildren();
  if (change.ShouldRecalcStyleFor(*this)) {
    child_change = RecalcOwnStyle(change);
    if (GetStyleChangeType() == kSubtreeStyleChange)
      child_change = child_change.ForceRecalcDescendants();
    ClearNeedsStyleRecalc();
  }

  if (child_change.TraversePseudoElements(*this)) {
    UpdatePseudoElement(kPseudoIdBackdrop, child_change);
    UpdatePseudoElement(kPseudoIdBefore, child_change);
  }

  if (child_change.TraverseChildren(*this)) {
    SelectorFilterParentScope filter_scope(*this);
    if (ShadowRoot* root = GetShadowRoot()) {
      if (child_change.TraverseChild(*root))
        root->RecalcStyle(child_change);
      RecalcDescendantStyles(StyleRecalcChange::kClearEnsured);
    } else {
      RecalcDescendantStyles(child_change);
    }
  }
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
void ContainerNode::RecalcDescendantStyles(const StyleRecalcChange change) {
  DCHECK(GetDocument().InStyleRecalc());
  DCHECK(!NeedsStyleRecalc());

  for (Node* child = firstChild(); child; child = child->nextSibling()) {
    if (!change.TraverseChild(*child))
      continue;
    if (child->IsTextNode())
      ToText(child)->RecalcTextStyle(change);
    else if (child->IsElementNode())
      ToElement(child)->RecalcStyle(change);
  }
}
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
void Element::RecalcStyle(const StyleRecalcChange change) {
  DCHECK(InActiveDocument());
  DCHECK(GetDocument().InStyleRecalc());
  DCHECK(!GetDocument().Lifecycle().InDetach());

  if (StyleRecalcBlockedByDisplayLock())
    return;

  if (HasCustomStyleCallbacks())
    WillRecalcStyle(change);

  StyleRecalcChange child_change = change.ForChildren();
  if (change.ShouldRecalcStyleFor(*this)) {
    child_change = RecalcOwnStyle(change);
    if (GetStyleChangeType() == kSubtreeStyleChange)
      child_change = child_change.ForceRecalcDescendants();
    ClearNeedsStyleRecalc();
  }

  if (child_change.TraversePseudoElements(*this)) {
    UpdatePseudoElement(kPseudoIdBackdrop, child_change);
    UpdatePseudoElement(kPseudoIdBefore, child_change);
  }

  if (child_change.TraverseChildren(*this)) {
    SelectorFilterParentScope filter_scope(*this);
    if (ShadowRoot* root = GetShadowRoot()) {
      if (child_change.TraverseChild(*root))
        root->RecalcStyle(child_change);
      RecalcDescendantStyles(StyleRecalcChange::kClearEnsured);
    } else {
      RecalcDescendantStyles(child_change);
    }
  }
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
StyleRecalcChange Element::RecalcOwnStyle(const StyleRecalcChange change) {
  DCHECK(GetDocument().InStyleRecalc());
  if (!CanParticipateInFlatTree()) {
    // This is a V0InsertionPoint. This whole block can be removed when Shadow
    // DOM V0 is removed.
    DCHECK(IsV0InsertionPoint());
    if (NeedsStyleRecalc())
      SetComputedStyle(nullptr);
    if (GetForceReattachLayoutTree())
      return change.ForceReattachLayoutTree();
    // Keep recalculating computed style for fallback children as if they were
    // children of the insertion point parent.
    return change;
  }

  if (change.RecalcChildren() && HasRareData() && NeedsStyleRecalc()) {
    // This element needs recalc because its parent changed inherited
    // properties or there was some style change in the ancestry which needed a
    // full subtree recalc. In that case we cannot use the BaseComputedStyle
    // optimization.
    if (ElementAnimations* element_animations =
            GetElementRareData()->GetElementAnimations())
      element_animations->SetAnimationStyleChange(false);
  }

  scoped_refptr<ComputedStyle> new_style;
  scoped_refptr<const ComputedStyle> old_style = GetComputedStyle();

  StyleRecalcChange child_change = change.ForChildren();

  // If we are on the find-in-page root, we need to calculate style for
  // invisible nodes in this subtree.
  if (!child_change.CalcInvisible() && this == GetDocument().FindInPageRoot())
    child_change = child_change.ForceCalcInvisible();

  if (ParentComputedStyle()) {
    if (old_style && change.IndependentInherit()) {
      // When propagating inherited changes, we don't need to do a full style
      // recalc if the only changed properties are independent. In this case, we
      // can simply clone the old ComputedStyle and set these directly.
      new_style = PropagateInheritedProperties();
    }
    if (!new_style)
      new_style = StyleForLayoutObject(child_change.CalcInvisible());
    if (new_style && !ShouldStoreComputedStyle(*new_style))
      new_style = nullptr;
  }
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
scoped_refptr<ComputedStyle> Element::StyleForLayoutObject(
    bool calc_invisible) {
  DCHECK(GetDocument().InStyleRecalc());

  // FIXME: Instead of clearing updates that may have been added from calls to
  // StyleForElement outside RecalcStyle, we should just never set them if we're
  // not inside RecalcStyle.
  if (ElementAnimations* element_animations = GetElementAnimations())
    element_animations->CssAnimations().ClearPendingUpdate();

  if (RuntimeEnabledFeatures::InvisibleDOMEnabled() &&
      hasAttribute(html_names::kInvisibleAttr) && !calc_invisible) {
    auto style =
        GetDocument().GetStyleResolver()->InitialStyleForElement(GetDocument());
    style->SetDisplay(EDisplay::kNone);
    return style;
  }

  scoped_refptr<ComputedStyle> style = HasCustomStyleCallbacks()
                                           ? CustomStyleForLayoutObject()
                                           : OriginalStyleForLayoutObject();

  if (!style) {
    DCHECK(IsPseudoElement());
    return nullptr;
  }
```

```
scoped_refptr<ComputedStyle> Element::OriginalStyleForLayoutObject() {
  return GetDocument().EnsureStyleResolver().StyleForElement(this);
}
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

Browser Default Style-> User style -> Author style 순으로 Matching해서 각 origin별 CSSPropertyValueSet 결과를 현재 element에 해당하는 ElementRuleCollector에 모아둔다.

ComputedStyle에 최종적으로 CSSPropertyValueSet을 저장한다.

```
scoped_refptr<ComputedStyle> StyleResolver::StyleForElement(
    Element* element,
    const ComputedStyle* default_parent,
    const ComputedStyle* default_layout_parent,
    RuleMatchingBehavior matching_behavior) {
  DCHECK(GetDocument().GetFrame());
  DCHECK(GetDocument().GetSettings());

                                        (.....)
  if (!base_computed_style) {
    GetDocument().GetStyleEngine().EnsureUAStyleForElement(*element);

    ElementRuleCollector collector(state.ElementContext(), selector_filter_,
                                   state.Style());

    MatchAllRules(state, collector,
                  matching_behavior != kMatchAllRulesExcludingSMIL);
                                        (.....)
    ApplyMatchedPropertiesAndCustomPropertyAnimations(
        state, collector.MatchedResult(), element);
    ApplyCallbackSelectors(state);
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
void StyleResolver::ApplyMatchedPropertiesAndCustomPropertyAnimations(
    StyleResolverState& state,
    const MatchResult& match_result,
    const Element* animating_element) {
  CacheSuccess cache_success = ApplyMatchedCache(state, match_result);
  NeedsApplyPass needs_apply_pass;
  if (!cache_success.IsFullCacheHit()) {
    ApplyCustomProperties(state, match_result, kExcludeAnimations,
                          cache_success, needs_apply_pass);
    ApplyMatchedAnimationProperties(state, match_result, cache_success,
                                    needs_apply_pass);
  }
  if (state.Style()->Animations() || state.Style()->Transitions() ||
      (animating_element && animating_element->HasAnimations())) {
    CalculateAnimationUpdate(state, animating_element);
    if (state.IsAnimatingCustomProperties()) {
      cache_success.SetFailed();
      ApplyCustomProperties(state, match_result, kIncludeAnimations,
                            cache_success, needs_apply_pass);
    }
  }
  if (!cache_success.IsFullCacheHit()) {
    ApplyMatchedStandardProperties(state, match_result, cache_success,
                                   needs_apply_pass);
  }
}
```

Style (4)

- CSS Spec에 정의된 CSS Style의 Origin에 따른 적용 우선순위에 따라 각 CSS Property를 적용한다.

Cascading order

The cascading algorithm determines how to find the value to apply for each property for each document element.

1. It first filters all the rules from the different sources to keep only the rules that apply to a given element. That means rules whose selector matches the given element and which are part of an appropriate media at-rule.
2. Then it sorts these rules according to their importance, that is, whether or not they are followed by `!important`, and by their origin. The cascade is in ascending order, which means that `!important` values from a user-defined style sheet have precedence over normal values originated from a user-agent style sheet:

	Origin	Importance
1	CSS Transitions	*see below*
2	user agent	normal
3	user	normal
	override 🗑	normal
4	author	normal
5	CSS Animations	*see below*
	override 🗑	`!important`
6	author	`!important`
7	user	`!important`
8	user agent	`!important`

3. In case of equality, the specificity of a value is considered to choose one or the other.

```
void StyleResolver::ApplyMatchedStandardProperties(
    StyleResolverState& state,
    const MatchResult& match_result,
    const CacheSuccess& cache_success,
    NeedsApplyPass& needs_apply_pass) {
  INCREMENT_STYLE_STATS_COUNTER(GetDocument().GetStyleEngine(),
                                matched_property_apply, 1);

  DCHECK(!cache_success.IsFullCacheHit());
  bool apply_inherited_only = cache_success.ShouldApplyInheritedOnly();

  // Now we have all of the matched rules in the appropriate order. Walk the
  // rules and apply high-priority properties first, i.e., those properties that
  // other properties depend on.  The order is (1) high-priority not important
  // (2) high-priority important, (3) normal not important and (4) normal
  // important.
  ApplyMatchedProperties<kHighPropertyPriority, kCheckNeedsApplyPass>(
      state, match_result.AllRules(), false, apply_inherited_only,
      needs_apply_pass);
  for (auto range : ImportantAuthorRanges(match_result)) {
    ApplyMatchedProperties<kHighPropertyPriority, kCheckNeedsApplyPass>(
        state, range, true, apply_inherited_only, needs_apply_pass);
  }
  for (auto range : ImportantUserRanges(match_result)) {
    ApplyMatchedProperties<kHighPropertyPriority, kCheckNeedsApplyPass>(
        state, range, true, apply_inherited_only, needs_apply_pass);
  }
  ApplyMatchedProperties<kHighPropertyPriority, kCheckNeedsApplyPass>(
      state, match_result.UaRules(), true, apply_inherited_only,
      needs_apply_pass);

                                (.....)
  // Now do the normal priority UA properties.
  ApplyMatchedProperties<kLowPropertyPriority, kCheckNeedsApplyPass>(
      state, match_result.UaRules(), false, apply_inherited_only,
      needs_apply_pass);

  // Cache the UA properties to pass them to LayoutTheme in
  // StyleAdjuster::AdjustComputedStyle.
  state.CacheUserAgentBorderAndBackground();

  // Now do the author and user normal priority properties and all the
  // !important properties.
  ApplyMatchedProperties<kLowPropertyPriority, kCheckNeedsApplyPass>(
      state, match_result.UserRules(), false, apply_inherited_only,
      needs_apply_pass);
  ApplyMatchedProperties<kLowPropertyPriority, kCheckNeedsApplyPass>(
      state, match_result.AuthorRules(), false, apply_inherited_only,
      needs_apply_pass);
  for (auto range : ImportantAuthorRanges(match_result)) {
    ApplyMatchedProperties<kLowPropertyPriority, kCheckNeedsApplyPass>(
        state, range, true, apply_inherited_only, needs_apply_pass);
  }
  for (auto range : ImportantUserRanges(match_result)) {
    ApplyMatchedProperties<kLowPropertyPriority, kCheckNeedsApplyPass>(
        state, range, true, apply_inherited_only, needs_apply_pass);
  }
  ApplyMatchedProperties<kLowPropertyPriority, kCheckNeedsApplyPass>(
      state, match_result.UaRules(), true, apply_inherited_only,
      needs_apply_pass);
```

Style (4)

- Style Recalc 수행때마다, StyleRule들을 토대로 각 Element에 최종적으로 적용되어야 하는 CSS Style을 계산해서 결과를 각각의 ComputedStyle에 저장.

```
template <CSSPropertyPriority priority,
          StyleResolver::ShouldUpdateNeedsApplyPass shouldUpdateNeedsApplyPass>
void StyleResolver::ApplyMatchedProperties(StyleResolverState& state,
                                           const MatchedPropertiesRange& range,
                                           bool is_important,
                                           bool inherited_only,
                                           NeedsApplyPass& needs_apply_pass) {
  if (range.IsEmpty())
    return;
                                (.....)
  for (const auto& matched_properties : range) {
    ApplyProperties<priority, shouldUpdateNeedsApplyPass>(
        state, matched_properties.properties.Get(), is_important,
        inherited_only, needs_apply_pass,
        static_cast<PropertyWhitelistType>(
            matched_properties.types_.whitelist_type));
  }
}
```

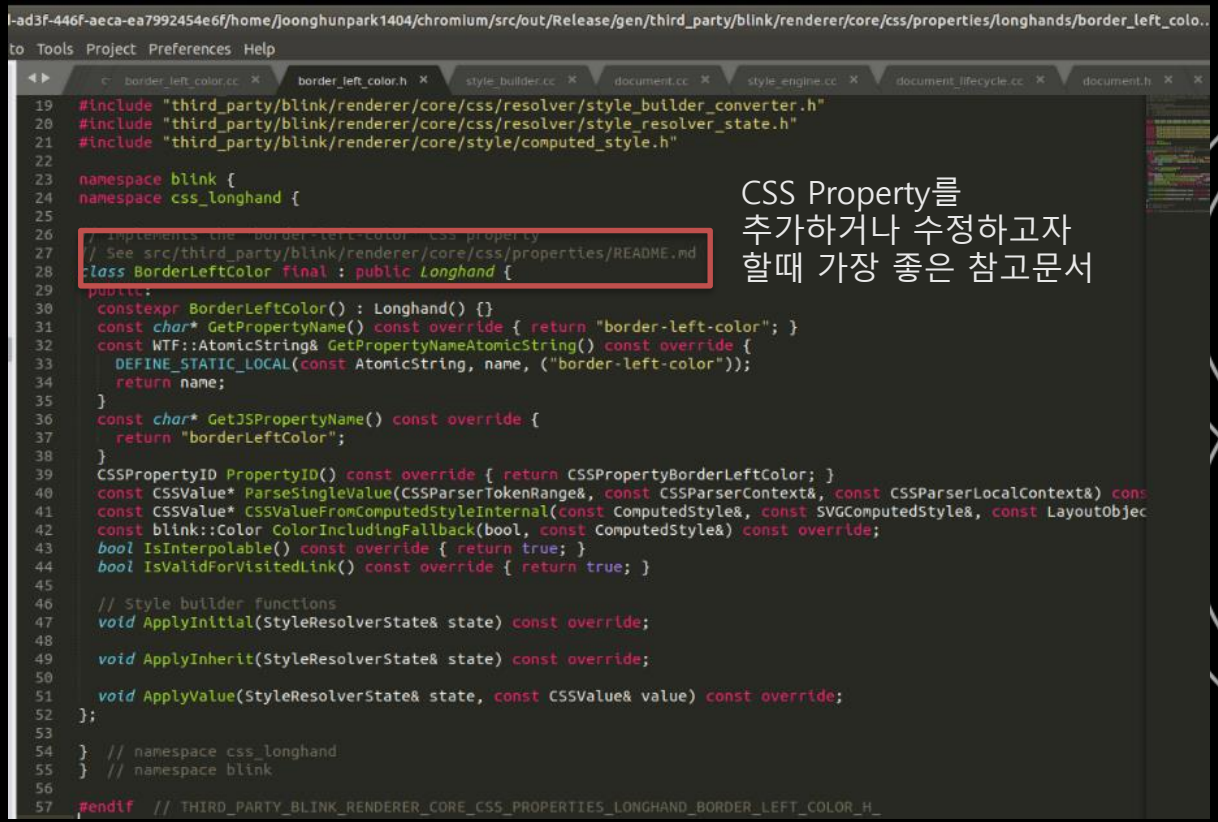
```
template <CSSPropertyPriority priority,
          StyleResolver::ShouldUpdateNeedsApplyPass shouldUpdateNeedsApplyPass>
void StyleResolver::ApplyProperties(
    StyleResolverState& state,
    const CSSPropertyValueSet* properties,
    bool is_important,
    bool inherited_only,
    NeedsApplyPass& needs_apply_pass,
    PropertyWhitelistType property_whitelist_type) {
                                (.....)
    if (!CSSPropertyPriorityData<priority>::PropertyHasPriority(property_id))
      continue;

    ApplyProperty<priority>(current, state);
  }
}
```

```
void StyleBuilder::ApplyProperty(const CSSProperty& property,
                                 StyleResolverState& state,
                                 const CSSValue& value) {
  DCHECK(!Variable::IsStaticInstance(property))
      << "Please use a CustomProperty instance to apply custom properties";
                                (.....)
  // CSSPropertyVariable currently handles initial/inherit inside ApplyValue.
  DCHECK(id != CSSPropertyVariable || !is_initial);
  DCHECK(id != CSSPropertyVariable || !is_inherit);

  if (is_initial)
    ToLonghand(property).ApplyInitial(state);
  else if (is_inherit)
    ToLonghand(property).ApplyInherit(state);
  else
    ToLonghand(property).ApplyValue(state, value);
}
```

```
template <CSSPropertyPriority priority>
static inline void ApplyProperty(
    const CSSPropertyValueSet::PropertyReference& reference,
    StyleResolverState& state) {
  static_assert(
      priority != kResolveVariables,
      "Application of custom properties must use specialized template");
  DCHECK_NE(reference.Id(), CSSPropertyVariable);
  StyleBuilder::ApplyProperty(reference.Property(), state, reference.Value());
}
```

Style (4)

- BorderLeftColor CSS Property의 예 (border_left_color.h)



Style (4)

- BorderLeftColor CSS Property의 예(border_left_color.cc)

ComputedStyle Object

```
// ../../third_party/blink/renderer/core/css/properties/css_property_methods.json5

#include "third_party/blink/renderer/core/css/properties/longhands/border_left_color.h"

#include "third_party/blink/renderer/core/css/css_primitive_value_mappings.h"
#include "third_party/blink/renderer/core/css/css_primitive_value_mappings.h"
#include "third_party/blink/renderer/core/css/resolver/style_builder_converter.h"
#include "third_party/blink/renderer/core/css/resolver/style_resolver_state.h"
#include "third_party/blink/renderer/core/style/computed_style.h"

namespace blink {
namespace css_longhand {

void BorderLeftColor::ApplyInitial(StyleResolverState& state) const {
  StyleColor color = StyleColor::CurrentColor();
  if (state.ApplyPropertyToRegularStyle())
    state.Style()->SetBorderLeftColor(color);
  if (state.ApplyPropertyToVisitedLinkStyle())
    state.Style()->SetVisitedLinkBorderLeftColor(color);
}

void BorderLeftColor::ApplyInherit(StyleResolverState& state) const {
  // Visited link style can never explicitly inherit from parent visite
  // style so no separate getters are needed.
  StyleColor color = state.ParentStyle()->BorderLeftColor();
  if (state.ApplyPropertyToRegularStyle())
    state.Style()->SetBorderLeftColor(color);
  if (state.ApplyPropertyToVisitedLinkStyle())
    state.Style()->SetVisitedLinkBorderLeftColor(color);
}

void BorderLeftColor::ApplyValue(StyleResolverState& state, const CSSValue& value) const {
  if (state.ApplyPropertyToRegularStyle())
    state.Style()->SetBorderLeftColor(StyleBuilderConverter::ConvertStyleColor(state, value));
  if (state.ApplyPropertyToVisitedLinkStyle()) {
    state.Style()->SetVisitedLinkBorderLeftColor(
        StyleBuilderConverter::ConvertStyleColor(state, value, true));
  }
}

}  // namespace css_longhand
}  // namespace blink
```

7.3. Explicit Defaulting

Several CSS-wide property values are defined below; declaring a property to have these values explicitly specifies a particular defaulting behavior. As specified in CSS Values and Units Level 3 [css-values-3], all CSS properties can accept these values.

7.3.1. Resetting a Property: the 'initial' keyword

If the cascaded value of a property is the *"initial"* keyword, the property's specified value is its initial value.

7.3.2. Explicit Inheritance: the 'inherit' keyword

If the cascaded value of a property is the *"inherit"* keyword, the property's specified and computed values are the inherited value.

https://drafts.csswg.org/css-cascade/#defaulting-keywords

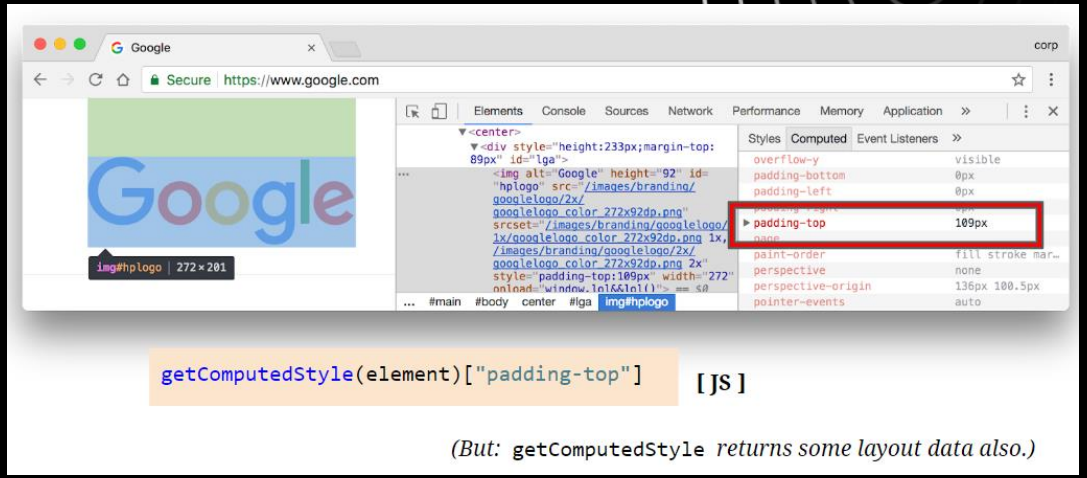
Style (5)

Resolved value – getComputedStyle()의 리턴값. computed value이거나 used value.

예를 들어, 부모 div의 width: 100px인 div { width: 20%; } 에 대해

- Specified value – 20%. Computed value is 'computed' from the specified value.
- Computed Value – 20%. CSS Property마다 type이 다름. (%, px, relative value, keyword, …)
- Used value - computed value에 모든 calculation을 적용한 후에 나오는 실제 layout에 사용되는 값. 이 경우 i.e. 20px.

Chromium Developer Tool – getComputedStyle()은 Blink의 ComputedStyle 객체의 정보를 리턴



Style(5)

Resolved value – getComputedStyle()의 리턴값. computed value이거나 used value.

Q. 그럼 resolved value는 언제 computed value이고 언제 used value인가?
A. 해당 내용을 지정하는 스펙이 있다. https://drafts.csswg.org/cssom/#resolved-values

§ 9. Resolved Values

`getComputedStyle()` was historically defined to return the "computed value" of an element or pseudo-element. However, the concept of "computed value" changed between revisions of CSS while the implementation of `getComputedStyle()` had to remain the same for compatibility with deployed scripts. To address this issue this specification introduces the concept of a ***resolved value***.

↳ 'color'
↳ 'line-height'
↳ 'outline-color'
↳ **A *resolved value special case property like 'color'* defined in another specification**
The resolved value is the used value.

↳ 'block-size'
↳ 'height'
↳ 'inline-size'
↳ 'margin-block-end'
↳ 'margin-block-start'
↳ 'margin-bottom'
↳ 'margin-inline-end'
↳ 'margin-inline-start'
↳ 'margin-left'
↳ 'margin-right'
↳ 'margin-top'
↳ 'padding-block-end'
↳ 'padding-block-start'
↳ 'padding-bottom'
↳ 'padding-inline-end'
↳ 'padding-inline-start'
↳ 'padding-left'
↳ 'padding-right'
↳ 'padding-top'
↳ 'width'
↳ **A *resolved value special case property like 'height'* defined in another specification**
If the property applies to the element or pseudo-element and the resolved value of the 'display' property is not 'none' or 'contents', then the resolved value is the used value. Otherwise the resolved value is the computed value.

Resolved value 관련 패치 예 (line-height: normal)

https://bugs.chromium.org/p/chromium/issues/detail?id=899489
이전에 언급한 스펙에 따라 line-height: normal은 getComputedStyle에 대해 resolved value로서 used value를 return해야 함.

Issue 899489: line-height: normal doesn't return the used value from getComputedStyle

Reported by emilio@chromium.org on Sat, Oct 27, 2018, 10:59 PM GMT+9 Project Member

Chrome Version : <Copy from: 'about:version'>
OS Version:
URLs (if applicable) : data:text/html, <script>document.write(getComputedStyle(document.querySelector("div")).lineHeight)</script>
Other browsers tested:
Add OK or FAIL after other browsers where you have tested this issue:
Safari: PASS
Firefox: PASS
IE/Edge: FAIL

What steps will reproduce the problem?
1. Open the URL above.

What is the expected result?
A pixel value.

What happens instead of that?
normal is serialized.

https://drafts.csswg.org/cssom/#resolved-values includes 'line-height' in the list of properties for:

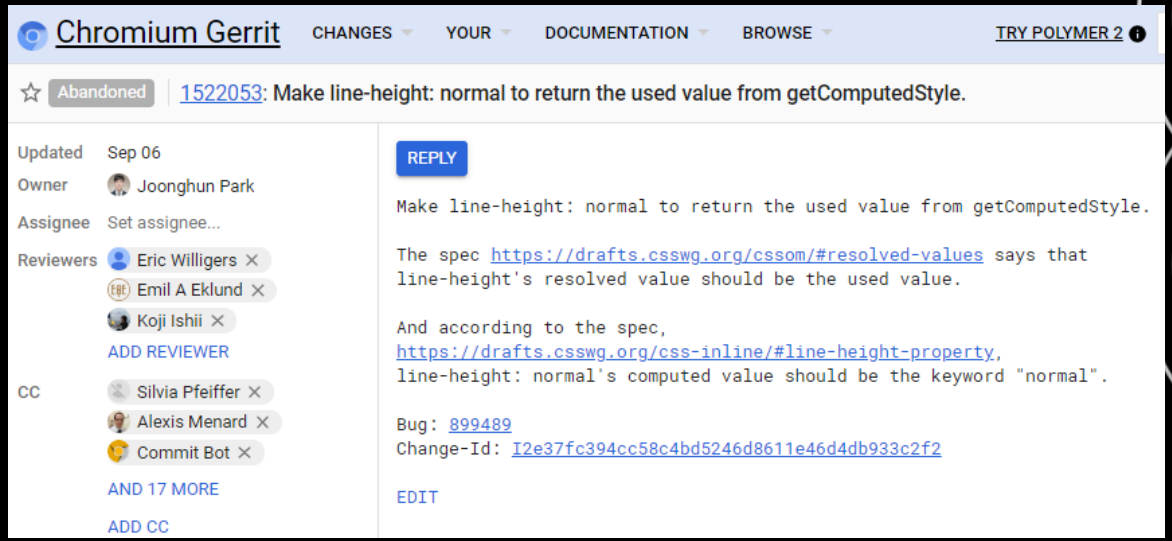
> The resolved value is the used value.

Resolved value 관련 패치 예 (line-height: normal)

- line-height: normal에 대한 당시 브라우저들의 동작

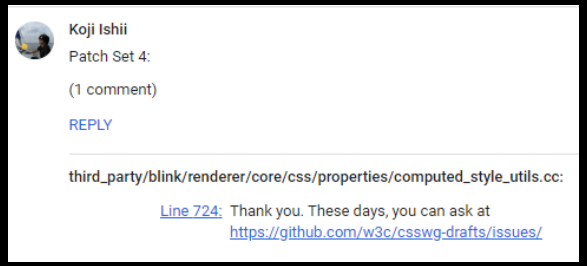
1) WebKit: used value 반환
2) Blink: css "normal" keyword 반환
3) FireFox: used value 반환

따라서 다른 브라우저들에 맞추기 위해 처음엔 Blink를 수정함



Resolved value 관련 패치 예 (line-height: normal)

- Chromium Owner인 Koji가 해당 패치의 스펙의 재확인을 요청.



Koji Ishii

Patch Set 4:

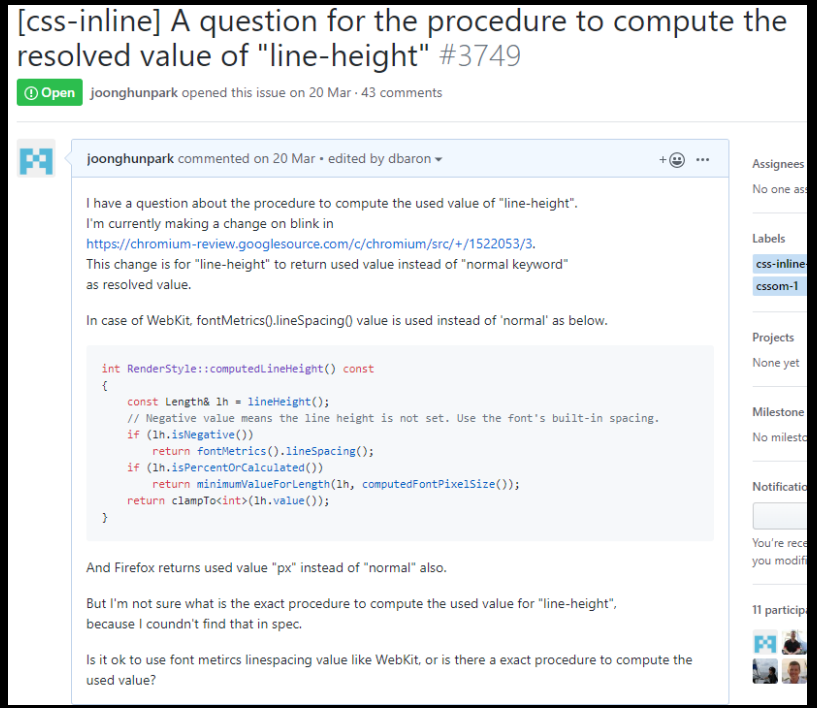
(1 comment)

REPLY

third_party/blink/renderer/core/css/properties/computed_style_utils.cc:

Line 724: Thank you. These days, you can ask at https://github.com/w3c/csswg-drafts/issues/

그에 따라 csswg에 line-height: normal에 대해 resolved value를 어떻게 할것인지에 대해 문의글을 작성.



[css-inline] A question for the procedure to compute the resolved value of "line-height" #3749

Open joonghunpark opened this issue on 20 Mar · 43 comments

joonghunpark commented on 20 Mar • edited by dbaron

I have a question about the procedure to compute the used value of "line-height".
I'm currently making a change on blink in
https://chromium-review.googlesource.com/c/chromium/src/+/1522053/3.
This change is for "line-height" to return used value instead of "normal keyword"
as resolved value.

In case of WebKit, fontMetrics().lineSpacing() value is used instead of 'normal' as below.

```
int RenderStyle::computedLineHeight() const
{
    const Length& lh = lineHeight();
    // Negative value means the line height is not set. Use the font's built-in spacing.
    if (lh.isNegative())
        return fontMetrics().lineSpacing();
    if (lh.isPercentOrCalculated())
        return minimumValueForLength(lh, computedFontPixelSize());
    return clampTo<int>(lh.value());
}
```

And Firefox returns used value "px" instead of "normal" also.

But I'm not sure what is the exact procedure to compute the used value for "line-height",
because I coundn't find that in spec.

Is it ok to use font metircs linespacing value like WebKit, or is there a exact procedure to compute the used value?

Assignees
No one ass

Labels
css-inline
cssom-1

Projects
None yet

Milestone
No milesto

Notificatio
You're rece
you modif

11 particip

Resolved value 관련 패치 예 (line-height: normal)

- 두달 반 정도의 discussion 끝에 해당 스펙을 변경하기로 결정.

- Line-height: normal에 대해서 getComputedStyle인 used value가 아니라 Computed value인 “normal” keyword를 return하기로 스펙을 변경.

- 당시 브라우저의 동작은 아래와 같았으므로,

1) WebKit: used value 반환
2) Blink: css “normal” keyword 반환
3) FireFox: used value 반환

- FireFox도 변경된 스펙에 맞게 동작을 변경하였으며,

WebKit의 경우 발표자가 동작을 변경(https://bugs.webkit.org/show_bug.cgi?id=201296)

Resolved value 관련 패치 예 (line-height: normal)

- WebKit의 해당 Patch

```
2019-09-09  Joonghun Park  <jh718.park@samsung.com>

        getComputedStyle for line-height: normal should return the keyword instead of a length
        https://bugs.webkit.org/show_bug.cgi?id=201296

        Reviewed by Ryosuke Niwa.

        Per https://github.com/w3c/csswg-drafts/issues/3749,
        Gecko and Blink has this behavior already.

        This patch makes WebKit has the same behavior with them.

        Tests: imported/w3c/web-platform-tests/css/css-inline/parsing/line-height-computed.html
               imported/w3c/web-platform-tests/css/cssom/getComputedStyle-line-height.html
               imported/w3c/web-platform-tests/html/rendering/replaced-elements/the-select-element/select-1-line-height.html

        * css/CSSComputedStyleDeclaration.cpp:
        (WebCore::lineHeightFromStyle):
```

해당 스펙 변경 후 현재 브라우저들의 동작

1) WebKit: css "normal" keyword 반환
2) Blink: css "normal" keyword 반환
3) FireFox: css "normal" keyword 반환

변경내역 - 간단

Source/WebCore/css/CSSComputedStyleDeclaration.cpp

side-by-side | header | annotate | revision log

expand: all | 20 below | 100 below

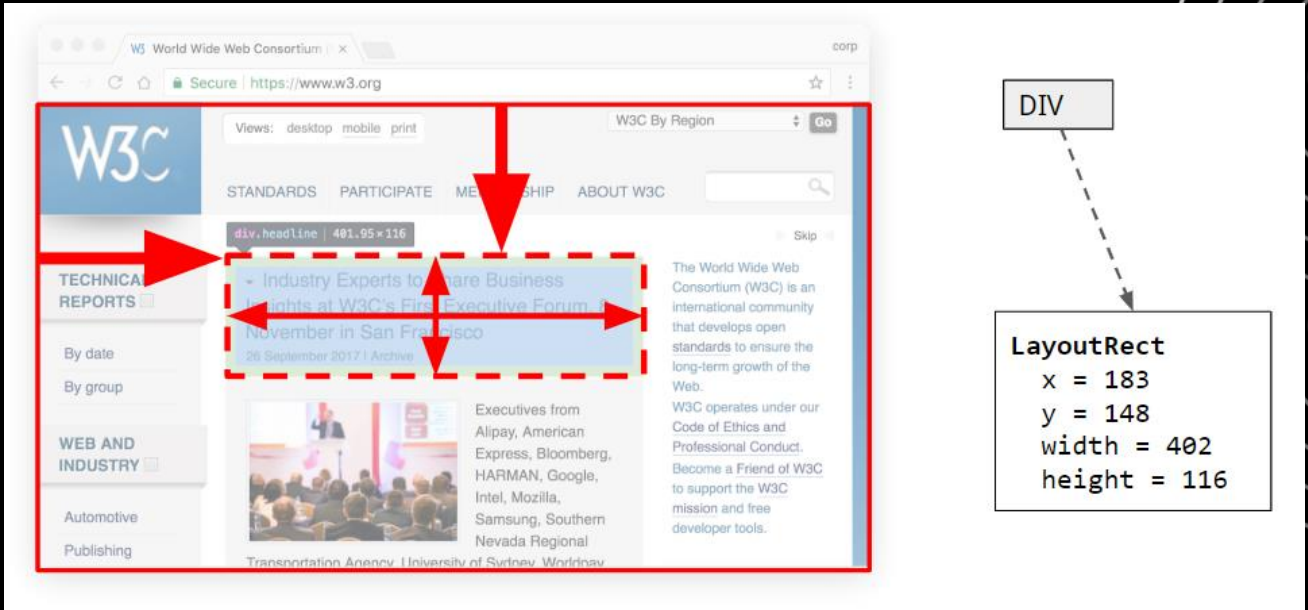
```
@    @    static Ref<CSSValue> fontFamilyFromStyle(const RenderStyle& style)
2022 2022 static Ref<CSSPrimitiveValue> lineHeightFromStyle(const RenderStyle& style)
2023 2023 {
2024 2024     Length length = style.lineHeight();
2025        -    if (length.isNegative()) // If true, line-height not set; use the font's line spacing.
2026        -        return zoomAdjustedPixelValue(style.fontMetrics().floatLineSpacing(), style);
     2025 +    if (length.isNegative())
     2026 +        return CSSValuePool::singleton().createIdentifierValue(CSSValueNormal);
2027 2027     if (length.isPercent()) {
2028 2028         // This is imperfect, because it doesn't include the zoom factor and the real computation
2029 2029         // for how high to be in pixels does include things like minimum font size and the zoom factor.
```

Browser 간 상호호환성의 중요성

- 브라우저 간에 interoperability를 확보하는 것은 웹 생태계에 있어 매우 중요
- Web Author들은 자신들의 컨텐츠가 자신들의 의도대로
  여러 브라우저에서 동일하게 보여지기를 기대
- 이러한 상호호환성이 확보되지 않은 CSS Property 등에 대해서 버그를 찾고
  해당 부분이 다른 브라우저와 같은 동작을 할 수 있도록 패치하는 것은
  웹엔진에 기여하는 데에 있어 좋은 진입점

Layout

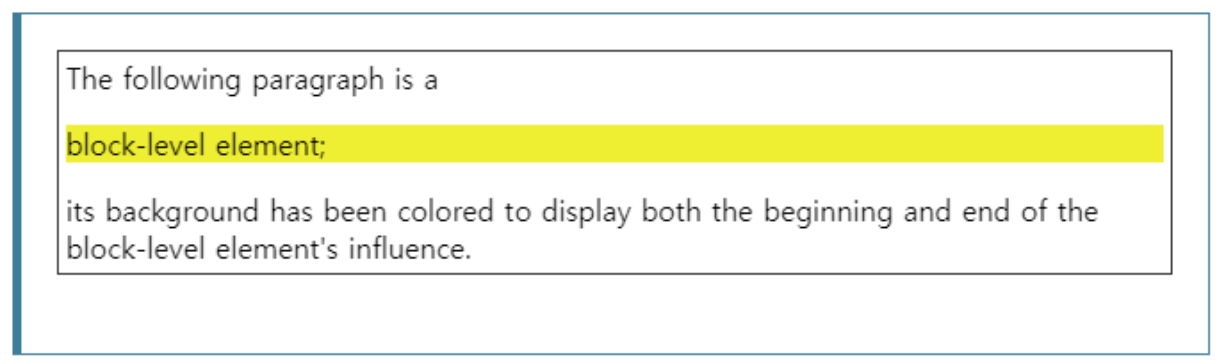
- DOM tree를 만들고 Style들을 계산한 후에는,
- 이 두가지를 결합해서 모든 Element들에 대해 시각적인 도형의 성질(geometry)을 결정한다(Layout). 즉, 위치(position)와 크기(size).
- 아래 그림의 경우, Block level Element에 대해서 이 Element가 점유하는 content area의 기하 영역(geometry region)에 대응하는 좌표와 크기를 계산(sizing)한다.



Block level Element

- Block Formatting Context에 참여하는 Element
- 새로운 Element는 Block 방향으로 Layout된다.
- 각각의 Element는 new line에서 시작.
- 아래는 <p>(paragraph) element의 code snippet example.

```
1 <div>The following paragraph is a <p class="highlight">block-level element;</p>
2 its background has been colored to display both the beginning and end of
3 the block-level element's influence.</div>
```



Inline level Element

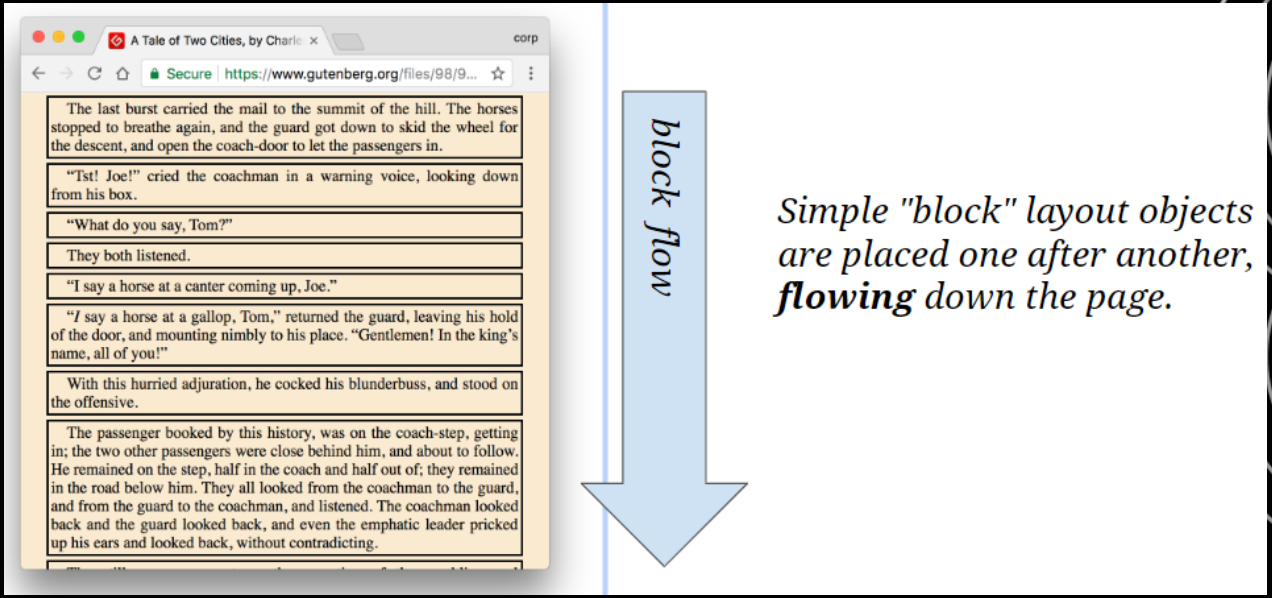
- Inline Formatting Context에 참여하는 Element
- 새로운 Element는 inline방향으로 Layout된다.
- 각각의 Element는 line 내의 어느 위치에서든 시작가능.
- 아래는 <p>(paragraph) element의 code snippet example.

```
<div>The following span is an <span class="highlight">inline element</span>;
its background has been colored to display both the beginning and end of
the inline element's influence.</div>
```

The following span is an inline element; its background has been colored to display both the beginning and end of the inline element's influence.

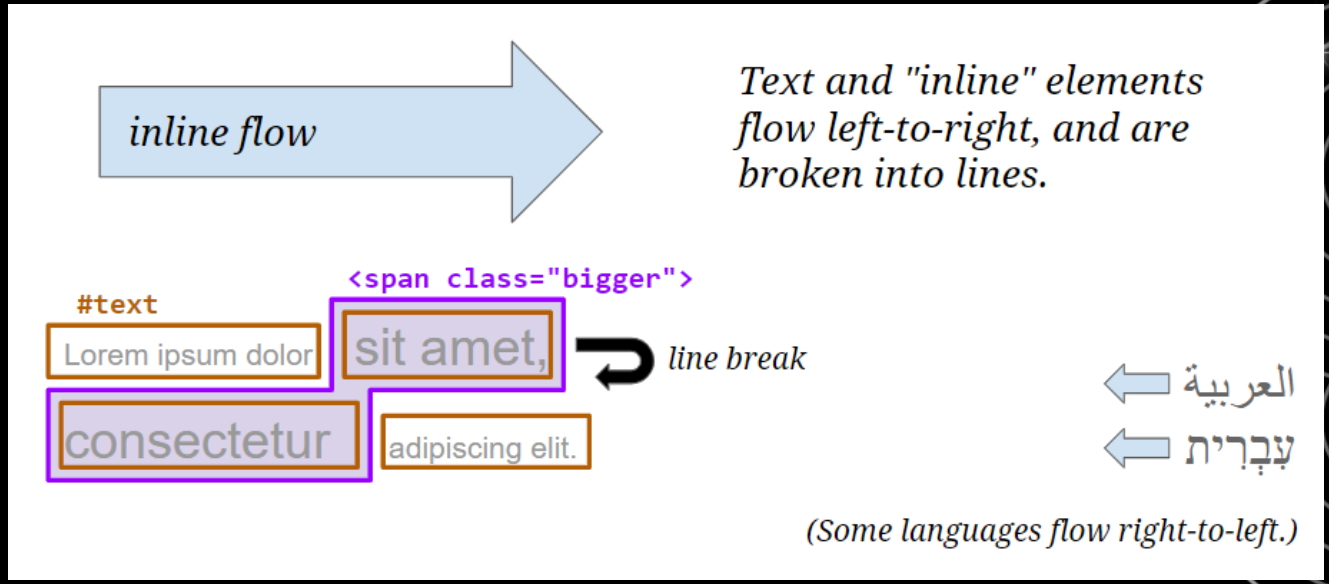
Block Flow

- Block level box들이 Layout시에 배치되는 방향.



Inline Flow

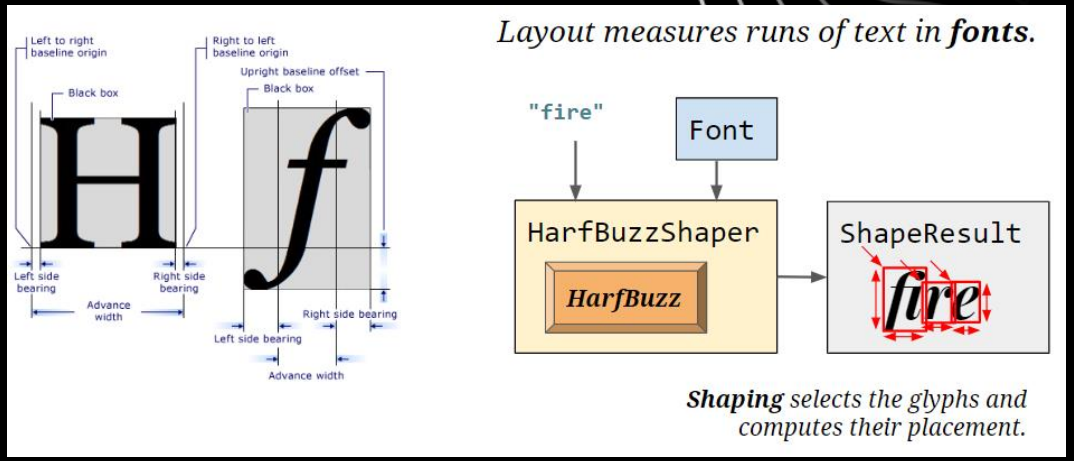
- Text와 Inline level box들이 Layout시에 배치되는 방향.



Shaping

- ComputedStyle에 적힌 Font를 적용해서 text에 대한 Glyph의 사이즈와 위치를 계산하는 과정
- Layout은 text의 크기를 font 정보(font metrics)를 통해 계산한다.
- Font metrics – 개별 font마다 가지고 있는 자신의 폰트 측정크기 정보
- LayoutText의 text 정보를 받아서 HarfBuzzShaper(다국어 지원 Shaping library)가 font 모양이 적용된 결과를 출력.
- 결과 text run의 Overall Width가 Layout에 사용되는 값이 된다.

- Grapheme – 자소
- Glyph – 활자
- Ligature – 합자
  (왼쪽 Shape의 fi가 하나로 합쳐진 Glyph)



CSS property 'Tab-size' 관련 패치 예

- https://drafts.csswg.org/css-text-3/#tab-size-property
- Tab-size property는 value로서

Number와 length type의 값을 받아들여야 한다.

4.2. Tab Character Size: the 'tab-size' property

Name:	***tab-size***
Value:	<integer> | <length>
Initial:	8
Applies to:	block containers
Inherited:	yes
Percentages:	N/A
Media:	visual
Computed value:	the specified integer or length made absolute
Animatable:	as length
Canonical order:	N/A

스펙 변경

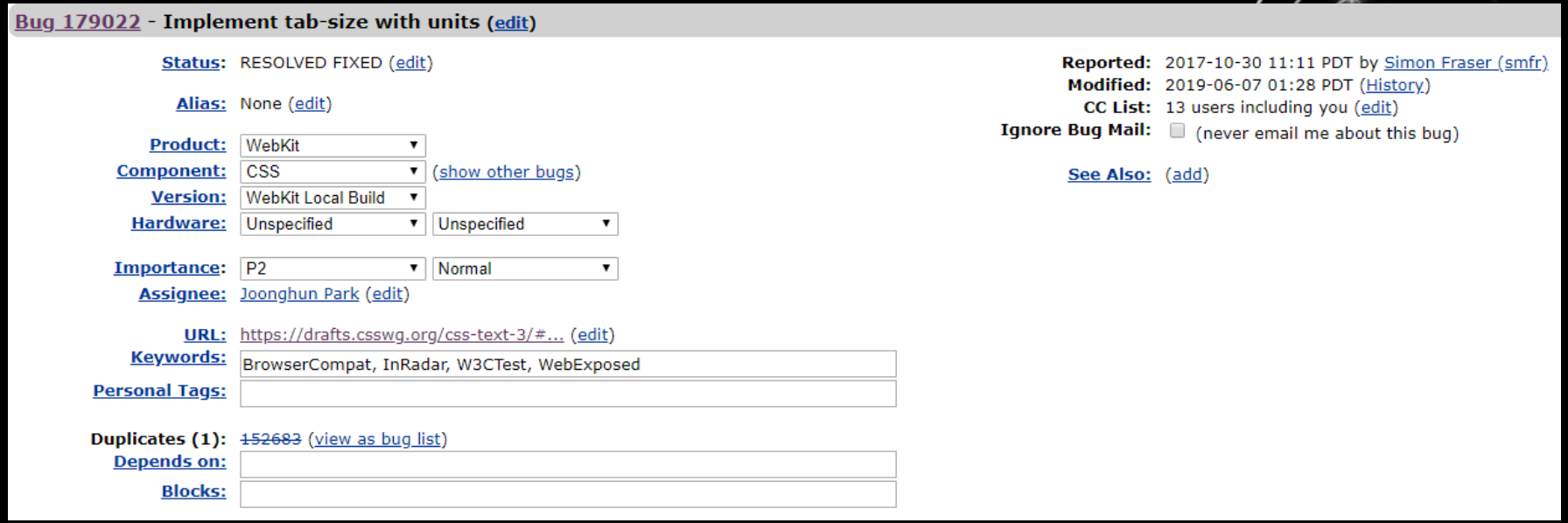
§ 4.2. Tab Character Size: the 'tab-size' property

Name:	***'tab-size'***
Value:	<number> | <length>
Initial:	8
Applies to:	block containers
Inherited:	yes
Percentages:	n/a
Computed value:	the specified number or absolute length
Canonical order:	n/a
Animation type:	by computed value type

예전 스펙: https://www.w3.org/TR/2013/WD-css-text-3-20131010/#tab-size-property

CSS property 'Tab-size' 관련 패치 예

- https://bugs.webkit.org/show_bug.cgi?id=179022
- WebKit의 경우 CSS type <integer> -> <number> 뿐 아니라 Length 지원 구현 역시 필요.



Bug 179022 - Implement tab-size with units (edit)

Status: RESOLVED FIXED (edit)

Alias: None (edit)

Product: WebKit
Component: CSS (show other bugs)
Version: WebKit Local Build
Hardware: Unspecified / Unspecified

Importance: P2 / Normal
Assignee: Joonghun Park (edit)

URL: https://drafts.csswg.org/css-text-3/#... (edit)
Keywords: BrowserCompat, InRadar, W3CTest, WebExposed
Personal Tags:

Duplicates (1): ~~152683~~ (view as bug list)
Depends on:
Blocks:

Reported: 2017-10-30 11:11 PDT by Simon Fraser (smfr)
Modified: 2019-06-07 01:28 PDT (History)
CC List: 13 users including you (edit)
Ignore Bug Mail: (never email me about this bug)

See Also: (add)

CSS property 'Tab-size' 관련 패치 예

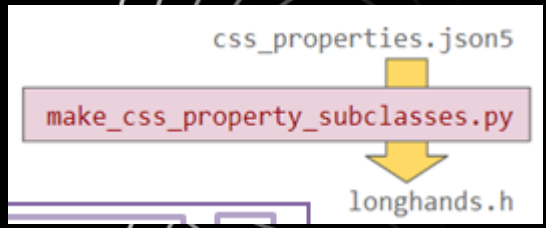
- WebCore/css/CSSProperties.json은 모든 CSS Property들의 리스트를 가지고 있다.
- 빌드 타임에 해당 파일 처리를 통해 필요한 파일들이 auto-generation된다.
- E.g. WebKitBuild/Release/DerivedSources/WebCore/StyleBuilder.cpp
- Blink의 css_properties.json5 파일에 해당

Source/WebCore/css/CSSProperties.json

ide-by-side | annotate | revision log

expand: all | 20 below | 100 below

```
3763 3763         },
3764 3764         "tab-size": {
3765 3765             "inherited": true,
     3766             "codegen-properties": {
     3767                 "converter": "TabSize"
     3768             },
3766 3769             "specification": {
3767 3770                 "category": "css-text",
3768                     "url": "https://www.w3.org/TR/css-text-3/#tab-size"
     3771                 "url": "https://drafts.csswg.org/css-text-3/#tab-size-property"
3769 3772             }
3770 3773         },
3771 3774         "text-align": {
```



CSS property 'Tab-size' 관련 패치 예

- Source/WebCore/platform/grahpics/TabSize.h를 추가

Tab-size : [property value];

1) Tab-size:8; 이면 탭 문자 하나가 space 8개
에 해당하는 Width임을 의미 (css integer type)a
2) Tab-size:8.5; 이면 탭 문자 하나가 space 8.5개
(css number type)
3) Tab-size: 30px; 이면 탭 문자 하나가 30px에 해당

이 Class를 통해 tab-size에 지정한 value type이
Number인지 length인지 구분할 수 있도록 함

```
29
30  #pragma once
31
32  namespace WebCore {
33
34  enum TabSizeValueType {
35      LengthValueType,
36      SpaceValueType,
37  };
38
39  struct TabSize {
40      TabSize(float numOrLength, TabSizeValueType isSpaces = SpaceValueType)
41          : m_value(numOrLength)
42          , m_isSpaces(isSpaces)
43      {
44      }
45
46      bool isSpaces() const
47      {
48          return m_isSpaces;
49      }
50
51      float widthInPixels(float spaceWidth) const
52      {
53          return m_isSpaces ? m_value * spaceWidth : m_value;
54      }
55
56      operator bool() const { return m_value; }
57
58      float m_value;
59      bool m_isSpaces;
60  };
61
62  inline bool operator==(const TabSize& a, const TabSize& b)
63  {
64      return (a.m_value == b.m_value) && (a.m_isSpaces == b.m_isSpaces);
65  }
66
67  inline bool operator!=(const TabSize& a, const TabSize& b)
68  {
69      return !(a == b);
70  }
71
72  } // namespace WebCore
```

CSS property 'Tab-size' 관련 패치 예

- Cmake나 Xcode에 TabSize.h를 추가

Gtk 등의 non-Apple port 빌드를 위해서는 cmake 파일에 해당 내역 추가

Source/WebCore/Headers.cmake

```
@ @ set(WebCore_PRIVATE_FRAMEWORK_HEADERS
1086 1086     platform/graphics/RemoteVideoSample.h
1087 1087     platform/graphics/RoundedRect.h
1088 1088     platform/graphics/StringTruncator.h
     1089     platform/graphics/TabSize.h
1089 1090     platform/graphics/TextRun.h
1090 1091     platform/graphics/TiledBacking.h
1091 1092     platform/graphics/TrackPrivateBase.h
```

Apple port 빌드를 위해서는 pbxproj에 추가. Xcode에서 추가하면 해당 파일에 아래 내용이 자동으로 생성됨

Source/WebCore/WebCore.xcodeproj/project.pbxproj

```
1021 1021 38F23AB31E8E83D000CE46F5 /* JSWebGPUComputeCommandEncoder.h in Headers */ = {isa = PBXBu
1022 1022 38F23AB51E8E83D000CE46F5 /* JSWebGPUComputePipelineState.h in Headers */ = {isa = PBXBui
1023 1023 3AC648B2129E146500C3EB25 /* EditingBoundary.h in Headers */ = {isa = PBXBuildFile; fileF
}; };
     1024 3BB6B81122A7D313003A2A69 /* TabSize.h in Headers */ = {isa = PBXBuildFile; fileRef = 3BB
1024 1025 3C244FEAA375AC633F88BE6F /* RenderLayerModelObject.h in Headers */ = {isa = PBXBuildFile
= (Private, ); }; };
1025 1026 3F2B33EF165AF15600E3987C /* JSWebKitCSSViewportRule.h in Headers */ = {isa = PBXBuildFil
1026 1027 3F42B31D1881191B00278AAC /* WebVideoFullscreenControllerAVKit.h in Headers */ = {isa = P
settings = {ATTRIBUTES = (Private, ); }; };

7165 7166 38F23AB11E8E83D000CE46F5 /* JSWebGPUComputePipelineState.cpp */ = {isa = PBXFileReferenc
JSWebGPUComputePipelineState.cpp; sourceTree = "<group>"; };
7166 7167 38F23AB21E8E83D000CE46F5 /* JSWebGPUComputePipelineState.h */ = {isa = PBXFileReference;
JSWebGPUComputePipelineState.h; sourceTree = "<group>"; };
7167 7168 3AC648B1129E146500C3EB25 /* EditingBoundary.h */ = {isa = PBXFileReference; fileEncoding
};
     7169 3BB6B80F22A7D311003A2A69 /* TabSize.h */ = {isa = PBXFileReference; fileEncoding = 4; la
7168 7170 3C244FE4A375AC633F88BE6F /* RenderLayerModelObject.h */ = {isa = PBXFileReference; fileE
= "<group>"; };
7169 7171 3C244FE5A375AC633F88BE6F /* RenderLayerModelObject.cpp */ = {isa = PBXFileReference; fil
sourceTree = "<group>"; };
7170 7172 3F2B33E3165ABD3500E3987C /* WebKitCSSViewportRule.idl */ = {isa = PBXFileReference; file
<group>"; };

24549 24551 849F77750EFEC62000908490 /* StrokeStyleApplier.h */,
24550 24552 087558C313B4A57D00F49307 /* SurrogatePairAwareTextIterator.cpp */,
24551 24553 087558C413B4A57D00F49307 /* SurrogatePairAwareTextIterator.h */,
      24554 3BB6B80F22A7D311003A2A69 /* TabSize.h */,
24552 24555 376DCCE013B4F966002EBEFC /* TextRun.cpp */,
24553 24556 A824B4640E2EF2EA0081A7B7 /* TextRun.h */,
24554 24557 CD1E7346167BC78E009A885D /* TextTrackRepresentation.cpp */,

31799 31802 0F03C0741884695E00A5F8CA /* SystemMemory.h in Headers */,
31800 31803 5D5975B319635F1100D00878 /* SystemVersion.h in Headers */,
31801 31804 A8CFF0510A154F09000A4234 /* TableLayout.h in Headers */,
      31805 3BB6B81122A7D313003A2A69 /* TabSize.h in Headers */,
31802 31806 463EB6231B8789E00096ED51 /* TagCollection.h in Headers */,
31803 31807 F55B3DD61251F12D003EF269 /* TelephoneInputType.h in Headers */,
31804 31808 7CC564B818BABEA6001B9652 /* TelephoneNumberDetector.h in Headers */,
```

CSS property 'Tab-size' 관련 패치 예

- CSS Parsing 단에서는 CSSProperty Parser가 TabSize를 CSS Integer가 아닌 CSS Number 또는 CSS Length로 Parsing할 수 있도록 변경

Source/WebCore/css/parser/CSSPropertyParser.cpp

side-by-side | header | annotate | revision log

expand: all | 20 below | 100 below

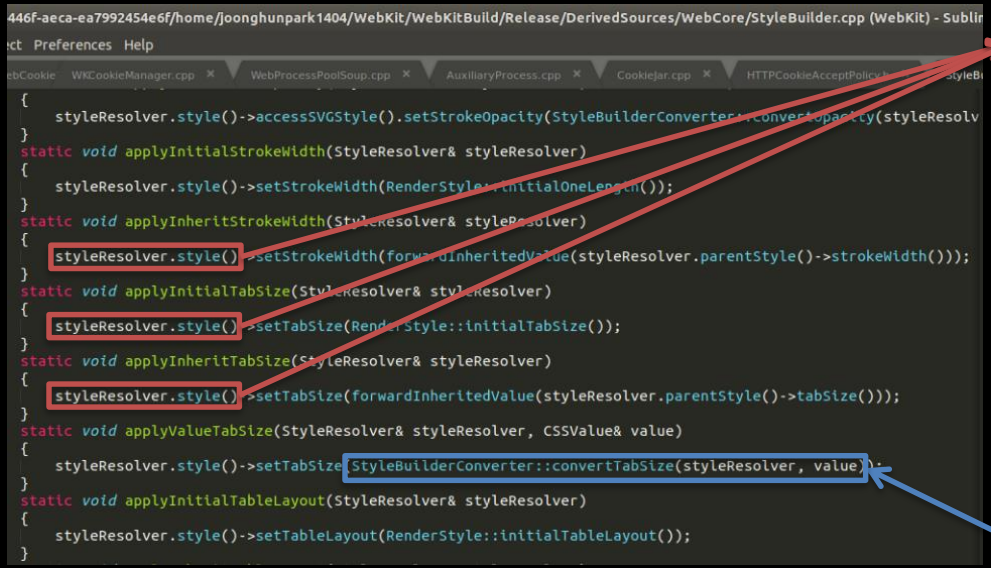
```
@    @    static RefPtr<CSSValue> consumeWordSpacing(CSSParserTokenRange& range, CSSParser
1121 1121     return consumeLengthOrPercent(range, cssParserMode, ValueRangeAll, UnitlessQuirk::Allow);
1122 1122 }
1123 1123
1124      static RefPtr<CSSValue> consumeTabSize(CSSParserTokenRange& range, CSSParserMode)
     1124 static RefPtr<CSSValue> consumeTabSize(CSSParserTokenRange& range, CSSParserMode cssParserMode)
1125 1125 {
1126      return consumeInteger(range, 0);
     1126 auto tabSize = consumeNumber(range, ValueRangeNonNegative);
     1127 if (tabSize)
     1128     return tabSize;
     1129 return consumeLength(range, cssParserMode, ValueRangeNonNegative);
1127 1130 }
1128 1131
1129 1132 #if ENABLE(TEXT_AUTOSIZING)
```

expand: 100 above | 20 above | all

CSS property 'Tab-size' 관련 패치 예

- WebKitBuild/Release/DerivedSources/WebCore/StyleBuilder.cpp

Build Time에 WebCore/css/CSSProperties.json를 처리해서 auto-generated 된 파일.



RenderStyle

WebKit의 RenderStyle은 Blink의 ComputedStyle에 해당

Source/WebCore/css/StyleBuilderConverter.h

side-by-side | annotate | revision log

expand: all | 20 below | 100 below

```
53 53 #include "Settings.h"
54 54 #include "StyleResolver.h"
55 55 #include "StyleScrollSnapPoints.h"
   56 #include "TabSize.h"
56 57 #include "TouchAction.h"
57 58 #include "TransformFunctions.h"
58 59 #include <wtf/Optional.h>
```

expand: 100 above | 20 above | all | 20 below | 100 below

```
@  @  public:
66 67     static Length convertLengthOrAuto(const StyleResolver&, const CSSValue&);
67 68     static Length convertLengthSizing(const StyleResolver&, const CSSValue&);
68 69     static Length convertLengthMaxSizing(const StyleResolver&, const CSSValue&);
   70     static TabSize convertTabSize(const StyleResolver&, const CSSValue&);
69 71     template<typename T> static T convertComputedLength(StyleResolver&, const CSSValue&);
70 72     template<typename T> static T convertLineWidth(StyleResolver&, const CSSValue&);
71 73     static float convertSpacing(StyleResolver&, const CSSValue&);
```

expand: 100 above | 20 above | all | 20 below | 100 below

```
@  @  inline Length StyleBuilderConverter::convertLengthMaxSizing(const StyleResolver&
255 257     return convertLengthSizing(styleResolver, value);
256 258 }
257 259
   260 inline TabSize StyleBuilderConverter::convertTabSize(const StyleResolver& styleResolver, const CSSValue& value)
   261 {
   262     auto& primitiveValue = downcast<CSSPrimitiveValue>(value);
   263     if (primitiveValue.isNumber())
   264         return TabSize(primitiveValue.floatValue(), SpaceValueType);
   265     return TabSize(primitiveValue.computeLength<float>(styleResolver.state().cssToLengthConversionData()), LengthValueType);
   266 }
   267
258 268 template<typename T>
259 269 inline T StyleBuilderConverter::convertComputedLength(StyleResolver& styleResolver, const CSSValue& value)
260 270 {
```

StyleBuilderConverter는 CSS System에서 사용되는 CSSValue를 Layout System에서 사용되는 Type(TabSize)으로 변환해서 RenderStyle에 저장

CSS property 'Tab-size' 관련 패치 예

- WebKit의 Web Inspector(Chromium의 Developer tool에 해당)에서 ComputedStyle을 조회했을 때 계산된 pixel 값을 반환하도록 CSSComputedStyleDeclaration 단을 수정

Source/WebCore/css/CSSComputedStyleDeclaration.cpp

```
@   @   RefPtr<CSSValue> ComputedStyleExtractor::valueForPropertyInStyle(const RenderSty
3026 3026               return cssValuePool.createIdentifierValue(CSSValueAuto);
3027 3027           return zoomAdjustedPixelValue(style.columnWidth(), style);
3028 3028       case CSSPropertyTabSize:
3029            -      return cssValuePool.createValue(style.tabSize(), CSSPrimitiveValue::CSS_NUMBER);
     3029       +      return cssValuePool.createValue(style.tabSize().widthInPixels(1.0), style.tabSize().isSpaces() ? CSSPrimitiveValue::CSS_NUMBER : CSSPrimitiveValue::CSS_PX);
3030 3030       case CSSPropertyCursor: {
3031 3031           RefPtr<CSSValueList> list;
3032 3032           auto* cursors = style.cursors();
```

CSS property 'Tab-size' 관련 패치 예

- Layout System 단의 수정
  스펙에 맞게 수정한대로 parsing된 tab-size property 값이 실제로 Layout시에 적용되도록 WebCore/platform/graphics/FontCascade.h 파일을 수정한다

https://drafts.csswg.org/css-text-3/#white-space-phase-2

Source/WebCore/platform/graphics/FontCascade.h

side-by-side | annotate | revision log

expand: all | 20 below | 100 below

```
@   @   public:
149 149
150 150     const FontMetrics& fontMetrics() const { return primaryFont().fontMetrics(); }
151 151     float spaceWidth() const { return primaryFont().spaceWidth() + m_letterSpacing; }
152         float tabWidth(const Font&, unsigned tabSize, float position) const;
153         float tabWidth(unsigned tabSize, float position) const { return tabWidth(primaryFont(), tabSize, position); }
    152     float tabWidth(const Font&, const TabSize&, float) const;
    153     float tabWidth(const TabSize& tabSize, float position) const { return tabWidth(primaryFont(), tabSize, position); }
154 154     bool hasValidAverageCharWidth() const;
155 155     bool fastAverageCharWidthIfAvailable(float &width) const; // returns true on success
156 156
```

expand: 100 above | 20 above | all | 20 below | 100 below

```
@   @   inline FontSelector* FontCascade::fontSelector() const
354 354     return m_fonts ? m_fonts->fontSelector() : nullptr;
355 355 }
356 356
357         inline float FontCascade::tabWidth(const Font& font, unsigned tabSize, float position)
    357     inline float FontCascade::tabWidth(const Font& font, const TabSize& tabSize, float pos
358 358 {
359         if (!tabSize)
    359     float baseTabWidth = tabSize.widthInPixels(font.spaceWidth());
    360     if (!baseTabWidth)
360 361         return letterSpacing();
361         float tabWidth = tabSize * font.spaceWidth() + letterSpacing();
362         float tabDeltaWidth = tabWidth - fmodf(position, tabWidth);
363         return (tabDeltaWidth < font.spaceWidth() / 2) ? tabWidth : tabDeltaWidth;
    362     float tabDeltaWidth = baseTabWidth - fmodf(position, baseTabWidth);
    363     return (tabDeltaWidth < font.spaceWidth() / 2) ? baseTabWidth : tabDeltaWidth;
364 364 }
365 365
366 366 }
```

expand: 100 above | 20 above | all

§ **4.1.3. Phase II: Trimming and Positioning**

As each line is laid out,

1. A sequence of collapsible spaces at the beginning of a line is removed.
2. If the tab size is zero, preserved tabs are not rendered. Otherwise, each preserved tab is rendered as a horizontal shift that lines up the start edge of the next glyph with the next tab stop. If this distance is less than 0.5ch, then the subsequent tab stop is used instead. ***Tab stops*** occur at points that are multiples of the tab size from the block's starting content edge. The tab size is given by the 'tab-size' property.

Note: See [UAX9] for rules on how U+0009 tabulation interacts with bidi.

CSS property 'Tab-size' 관련 패치 예

SimpleLineLaytoutTextFragmentIterator.h, RenderStyle.h 등
다른 파일들은 tab-size를
Unsigned integer로만 취급하던
상태에서
TabSize를 도입함에 따른
변경사항들에 불과

WebKit의 RenderStyle은
Blink의 ComputedStyle에 해당

Source/WebCore/rendering/SimpleLineLayoutTextFragmentIterator.h

side-by-side | annotate | revision log

expand: all | 20 below | 100 below

```
@ @ public:
118 118         bool breakNBSP;
119 119         bool keepAllWordsForCJK;
120 120         float wordSpacing;
121             unsigned tabWidth;
    121         TabSize tabWidth;
122 122         bool shouldHyphenate;
123 123         float hyphenStringWidth;
124 124         unsigned hyphenLimitBefore;
```

expand: 100 above | 20 above | all

Source/WebCore/rendering/style/RenderStyle.h

```
@ @ public:
661 661     TextCombine textCombine() const { return static_cast<TextCombine>(m_rareNonInheritedData->textCombine); }
662 662     bool hasTextCombine() const { return textCombine() != TextCombine::None; }
663 663
664         unsigned tabSize() const { return m_rareInheritedData->tabSize; }
    664     const TabSize& tabSize() const { return m_rareInheritedData->tabSize; }
665 665
666 666     // End CSS3 Getters
667 667
```

```
@ @ public:
1190 1190     void setBackdropFilter(const FilterOperations& ops) { SET_NESTED_VAR(m_rareNonInheritedData, backdropFilter, operations, ops); }
1191 1191 #endif
1192 1192
1193             void setTabSize(unsigned size) { SET_VAR(m_rareInheritedData, tabSize, size); }
     1193     void setTabSize(TabSize size) { SET_VAR(m_rareInheritedData, tabSize, size); }
1194 1194
1195 1195     void setBreakBefore(BreakBetween breakBehavior) { SET_VAR(m_rareNonInheritedData, breakBefore, static_cast<unsigned>(breakBehavior)); }
1196 1196     void setBreakAfter(BreakBetween breakBehavior) { SET_VAR(m_rareNonInheritedData, breakAfter, static_cast<unsigned>(breakBehavior)); }
```

```
@ @ public:
1672 1672     static GridPosition initialGridItemRowStart() { return GridPosition(); }
1673 1673     static GridPosition initialGridItemRowEnd() { return GridPosition(); }
1674 1674
1675             static unsigned initialTabSize() { return 8; }
     1675     static TabSize initialTabSize() { return 8; }
1676 1676
1677 1677     static const AtomicString& initialLineGrid() { return nullAtom(); }
1678 1678     static LineSnap initialLineSnap() { return LineSnap::None; }
```

CSS property 'Tab-size' 관련 패치 예

TextRun.h, TextRun.cpp 등의 파일 이름에서 알 수 있듯이 웹엔진 구현은 Spec에 있는 개념의 이름을 그대로 가져오는 등 스펙의 구현내용을 잘 이해할 수 있도록 쓰여있다.

Source/WebCore/platform/graphics/TextRun.cpp

side-by-side | header | annotate | revision log

expand: all | 20 below | 100 below

```
@   @   namespace WebCore {
30  30
31  31  struct ExpectedTextRunSize {
32  32      String text;
33          unsigned integer1;
    33      TabSize tabSize;
34  34      float float1;
35  35      float float2;
36  36      float float3;
```

expand: 100 above | 20 above | all

Source/WebCore/platform/graphics/TextRun.h

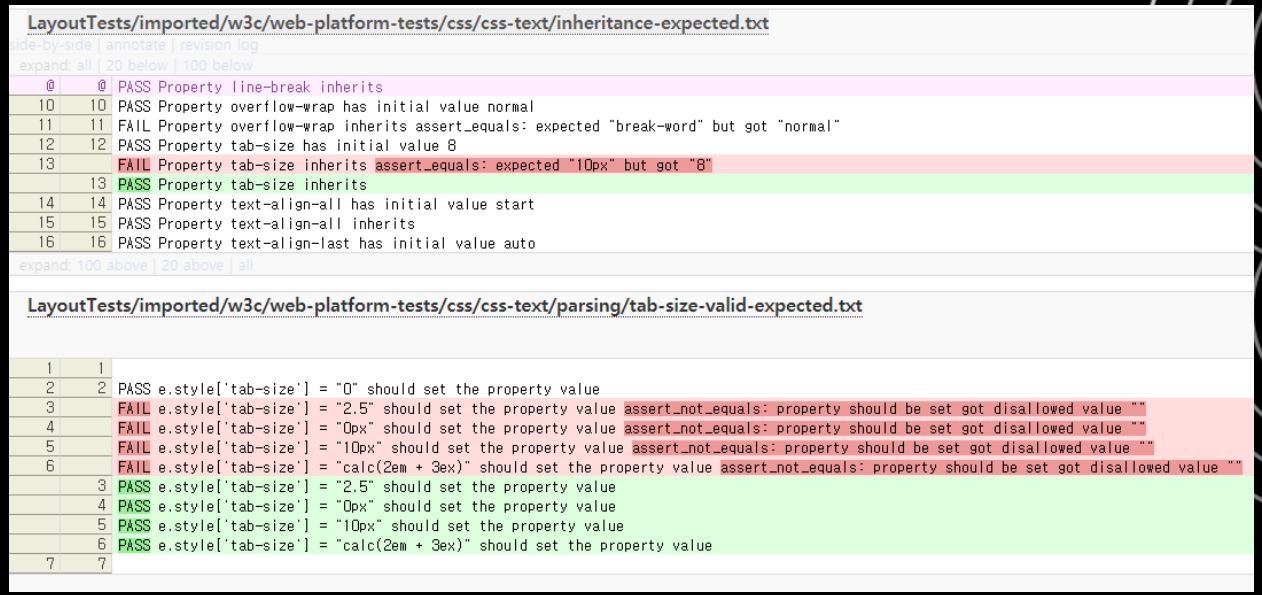
```
24  24  #ifndef TextRun_h
25  25  #define TextRun_h
26  26
    27  #include "TabSize.h"
27  28  #include "TextFlags.h"
28  29  #include "WritingMode.h"
29  30  #include <wtf/text/StringView.h>

@   @   public:
95  96      void setHorizontalGlyphStretch(float scale) { m_horizontalGlyphStretch = scale; }
96  97
97  98      bool allowTabs() const { return m_allowTabs; }
98          unsigned tabSize() const { return m_tabSize; }
99          void setTabSize(bool, unsigned);
    99      const TabSize& tabSize() const { return m_tabSize; }
    100     void setTabSize(bool, const TabSize&);
100 101
101 102       float xPos() const { return m_xpos; }
102 103       void setXPos(float xPos) { m_xpos = xPos; }

@   @   public:
118 119 private:
119 120     String m_text;
120 121
121         unsigned m_tabSize;
    122     TabSize m_tabSize;
122 123
```

CSS property 'Tab-size' 관련 패치 예

- Fail나던 Test들이 Pass로 변경되는 모습

wpt – web platform tests(https://github.com/web-platform-tests/wpt)
웹 표준을 테스트하는 테스트케이스. 모든 브라우저에서 공통으로 사용하는 테스트.

- 기본적으로 패치들은 수정사항이 의도대로 동작하는지 확인할 수 있도록 테스트가 필요



LayoutTests/imported/w3c/web-platform-tests/css/css-text/inheritance-expected.txt

side-by-side | annotate | revision log

expand: all | 20 below | 100 below

```
@ @ PASS Property line-break inherits
10 10 PASS Property overflow-wrap has initial value normal
11 11 FAIL Property overflow-wrap inherits assert_equals: expected "break-word" but got "normal"
12 12 PASS Property tab-size has initial value 8
13    FAIL Property tab-size inherits assert_equals: expected "10px" but got "8"
   13 PASS Property tab-size inherits
14 14 PASS Property text-align-all has initial value start
15 15 PASS Property text-align-all inherits
16 16 PASS Property text-align-last has initial value auto
```

expand: 100 above | 20 above | all

LayoutTests/imported/w3c/web-platform-tests/css/css-text/parsing/tab-size-valid-expected.txt

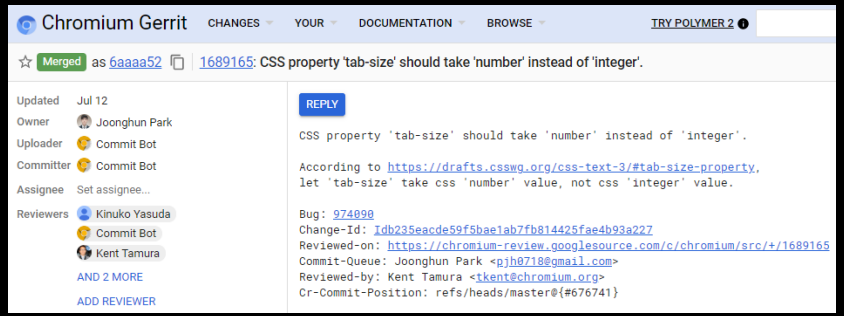
```
1 1
2 2 PASS e.style['tab-size'] = "0" should set the property value
3   FAIL e.style['tab-size'] = "2.5" should set the property value assert_not_equals: property should be set got disallowed value ""
4   FAIL e.style['tab-size'] = "0px" should set the property value assert_not_equals: property should be set got disallowed value ""
5   FAIL e.style['tab-size'] = "10px" should set the property value assert_not_equals: property should be set got disallowed value ""
6   FAIL e.style['tab-size'] = "calc(2em + 3ex)" should set the property value assert_not_equals: property should be set got disallowed value ""
  3 PASS e.style['tab-size'] = "2.5" should set the property value
  4 PASS e.style['tab-size'] = "0px" should set the property value
  5 PASS e.style['tab-size'] = "10px" should set the property value
  6 PASS e.style['tab-size'] = "calc(2em + 3ex)" should set the property value
7 7
```

CSS property 'Tab-size' 관련 패치 예

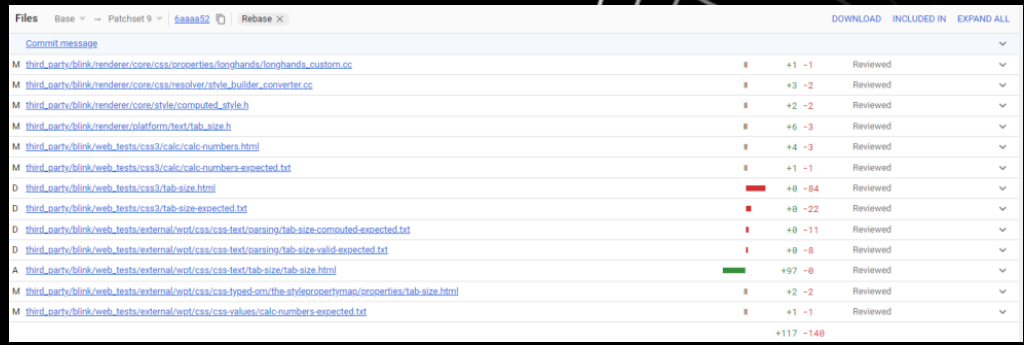
- WebKit과 Blink는 기본적으로 유사

Blink가 WebKit에서 Forking된 프로젝트이기 때문에
한쪽에 커밋한 패치는 다른 쪽에 포팅이 용이하다.

그러므로 아래와 같이 WebKit 패치를 Blink에 포팅하고 Merge함.



변경내역



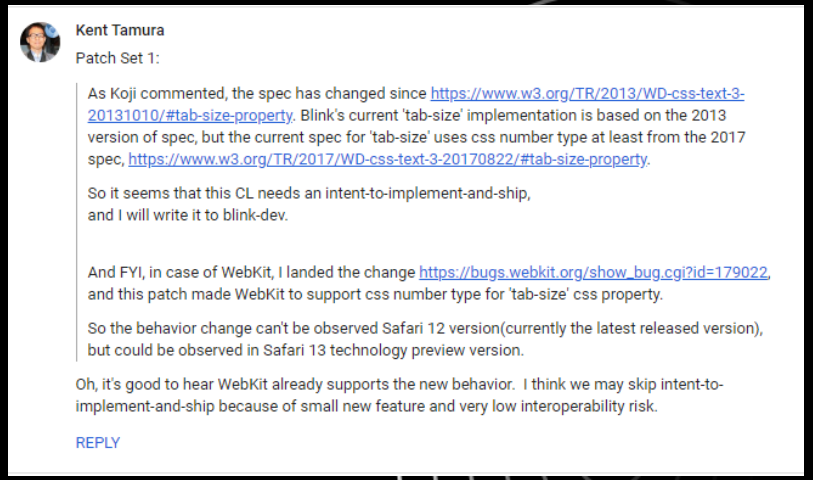
CSS property 'Tab-size' 관련 패치 예

- Chromium의 경우

새로운 Feature에 대해서는
Intent to implement and Ship이라고 하는
폼을 만들어서 blink-dev mailing list에
보내고 해당 API의 Owner들 3명 이상에게
LGTM을 받아야 해당 내용을 Shipping할 수 있다.

- 이 CL(Change List)의 경우

WebKit에 발표자가 이미 Shipping했기 때문에
Interoperability risk가 낮아서
이 단계를 건너뛰어도 좋다고 Kent Tamura에게
Confirm을 받고 바로 review 받은 후 merge함.



Kent Tamura

Patch Set 1:

> As Koji commented, the spec has changed since https://www.w3.org/TR/2013/WD-css-text-3-20131010/#tab-size-property. Blink's current 'tab-size' implementation is based on the 2013 version of spec, but the current spec for 'tab-size' uses css number type at least from the 2017 spec, https://www.w3.org/TR/2017/WD-css-text-3-20170822/#tab-size-property.
>
> So it seems that this CL needs an intent-to-implement-and-ship,
> and I will write it to blink-dev.
>
> And FYI, in case of WebKit, I landed the change https://bugs.webkit.org/show_bug.cgi?id=179022, and this patch made WebKit to support css number type for 'tab-size' css property.
>
> So the behavior change can't be observed Safari 12 version(currently the latest released version), but could be observed in Safari 13 technology preview version.

Oh, it's good to hear WebKit already supports the new behavior. I think we may skip intent-to-implement-and-ship because of small new feature and very low interoperability risk.

REPLY

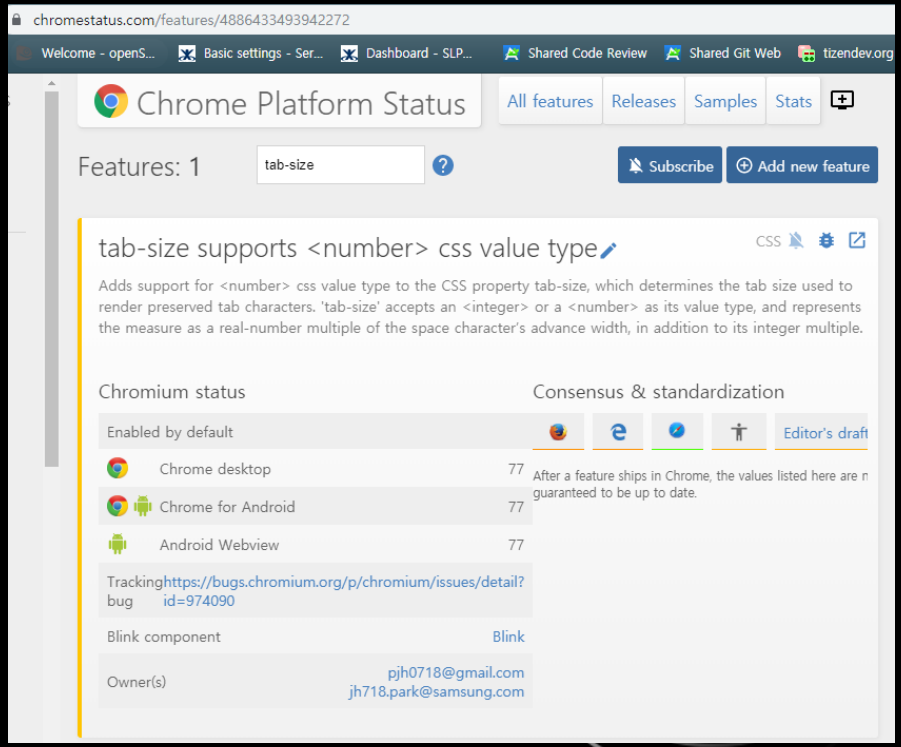
CSS property 'Tab-size' 관련 패치 예

- Chromium의 경우

Web developer들에게 새로운 feature나 동작을 알리기 위해 이러한 변경점들은 Chrome Platform Status에 추가하도록 함

- 이에 따라

해당 패치에 대한 내용을 Chrome Platform Status에 엔트리로 추가



결론

- CSSWG의 논의에 참여하고, WebKit과 Blink에 패치를 넣으며 Contribution하는 과정에서 CSS, Layout에 대해 공부하게 되고 더 잘 이해할 수 있게 된다.

- Bugs.chromium.org, bugs.webkit.org에서 본인이 관심있는 버그를 찾은 후에 해당 버그를 수정해서 리뷰를 받고 커밋할 수 있다.

- 본인이 수정한 버그 또는 본인이 도입한 새로운 Feature를 Public이 사용하는 것에서 오는 보람이 있다.

Castanets Project

- https://github.com/Samsung/Castanets

Browser process와 Renderer Process를 서로 다른 device에 실행해서
성능 향상과 메모리 절약을 목표로 하는 오픈소스 프로젝트

THANK YOU

SOSCON2019

SAMSUNG OPEN SOURCE CONFERENCE 2019