Extending your App with Safari App Extensions

Session 214

Brian Weinstein Safari Engineer
Damian Kaleta Safari Engineer
Zach Li Safari Engineer

Safari App Extensions

Capabilities

NEW

Customize web pages

Safari App Extensions

Capabilities

NEW

Customize web pages

Block loading of resources or elements

Safari App Extensions

NEW

Capabilities

Customize web pages

Block loading of resources or elements

Add toolbar buttons

Safari App Extensions

NEW

Capabilities

Customize web pages

Block loading of resources or elements

Add toolbar buttons

Display popovers

Safari App Extensions

NEW

Capabilities

Customize web pages

Block loading of resources or elements

Add toolbar buttons

Display popovers

Add items to context menus on webpages

Safari App Extensions

Benefits - Built Using App Extensions

Developed using Xcode

Safari App Extensions

Benefits - Built Using App Extensions

Developed using Xcode

Run native code

Safari App Extensions

Benefits - Distributed with your app

Sold through the Mac App Store

Safari App Extensions

Benefits - Distributed with your app

Sold through the Mac App Store

Same version as your app

Safari App Extensions

Types

Safari App Extensions

Types

Content Blockers

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Content Blockers

Zach Li Safari Engineer

and many more…

Content Blockers

Bring an iOS content blocker to macOS

Content Blockers

Bring an iOS content blocker to macOS

Safari Extensibility: Content Blocking and Shared Links

WWDC 2015

my-food-blog.com

my food blog

Apple Pie

Pumpkin Salad

Red Velvet Cake

Beef Steak

Chocolate Chip Cookies

Soup Dumplings

Demo

Dessert Blocker

Content Blockers

Content Blockers

New API

NEW

```
// SFContentBlockerManager
public class SFContentBlockerManager : NSObject {
   public class func getStateOfContentBlocker(identifier identifier: String,
      completionHandler: (SFContentBlockerState?, NSError?) -> Void)
}

// SFContentBlockerState
public class SFContentBlockerState : NSObject {
   public var isEnabled: Bool { get }
}
```

NEW

```
// SFContentBlockerManager
public class SFContentBlockerManager : NSObject {
    public class func getStateOfContentBlocker(identifier identifier: String,
        completionHandler: (SFContentBlockerState?, NSError?) -> Void)
}

// SFContentBlockerState
public class SFContentBlockerState : NSObject {
    public var isEnabled: Bool { get }
}
```

Check Safari Services APIs Availability

NEW

SFSafariServicesAvailable()

Safari Services APIs will be available on:

- macOS Sierra
- OS X El Capitan with Safari 10 installed

Check Safari Services APIs Availability

NEW

SFSafariServicesAvailable()

Safari Services APIs will be available on:

- macOS Sierra
- OS X El Capitan with Safari 10 installed

```
import SafariServices
if SFSafariServicesAvailable() {
   SFContentBlockerManager.getStateOfContentBlocker(identifier:
      "com.apple.DessertBlocker.DessertBlockerExtension", completionHandler: {
         (state, error) in
            // Check `state` to find if the content blocker is enabled.
         })
} else {
   // The API is not available, have a fallback behavior.
}
```

Check Safari Services APIs Availability

NEW

SFSafariServicesAvailable()

Safari Services APIs will be available on:

- macOS Sierra
- OS X El Capitan with Safari 10 installed

```
import SafariServices
if SFSafariServicesAvailable() {
   SFContentBlockerManager.getStateOfContentBlocker(identifier:
      "com.apple.DessertBlocker.DessertBlockerExtension", completionHandler: {
         (state, error) in
            // Check `state` to find if the content blocker is enabled.
         })
} else {
   // The API is not available, have a fallback behavior.
}
```

Check Safari Services APIs Availability

NEW

SFSafariServicesAvailable()

Safari Services APIs will be available on:

- macOS Sierra
- OS X El Capitan with Safari 10 installed

```
import SafariServices
if SFSafariServicesAvailable() {
   SFContentBlockerManager.getStateOfContentBlocker(identifier:
      "com.apple.DessertBlocker.DessertBlockerExtension", completionHandler: {
         (state, error) in
            // Check `state` to find if the content blocker is enabled.
         })
} else {
   // The API is not available, have a fallback behavior.
}
```

Check Safari Services APIs Availability

NEW

SFSafariServicesAvailable()

Safari Services APIs will be available on:

- macOS Sierra
- OS X El Capitan with Safari 10 installed

```
import SafariServices
if SFSafariServicesAvailable() {
   SFContentBlockerManager.getStateOfContentBlocker(identifier:
      "com.apple.DessertBlocker.DessertBlockerExtension", completionHandler: {
         (state, error) in
            // Check `state` to find if the content blocker is enabled.
         })
} else {
   // The API is not available, have a fallback behavior.
}
```

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

```
// Adding a style sheet - Info.plist

<key>NSExtension</key>
<dict>
   <key>SFSafariStyleSheet</key>
   <array>
      <dict>
         <key>Style Sheet</key>
         <string>style.css</string>
      </dict>
   </array>
</dict>
```

```
// Adding a style sheet - Info.plist

<key>NSExtension</key>
<dict>
    <key>SFSafariStyleSheet</key>
    <array>
        <dict>
            <key>Style Sheet</key>
            <string>style.css</string>
        </dict>
    </array>
</dict>
```

```
// Adding a style sheet - Info.plist

<key>NSExtension</key>
<dict>
   <key>SFSafariStyleSheet</key>
   <array>
      <dict>
         <key>Style Sheet</key>
         <string>style.css</string>
      </dict>
   </array>
</dict>
```

```
// Adding a style sheet - Info.plist

<key>NSExtension</key>
<dict>
   <key>SFSafariStyleSheet</key>
   <array>
      <dict>
         <key>Style Sheet</key>
         <string>style.css</string>
      </dict>
   </array>
</dict>
```

```
// Adding a content script - Info.plist

<key>NSExtension</key>
<dict>
   <key>SFSafariContentScript</key>
   <array>
      <dict>
         <key>Script</key>
         <string>replace.js</string>
      </dict>
   </array>
</dict>
```

```
// Adding a content script - Info.plist

<key>NSExtension</key>
<dict>
    <key>SFSafariContentScript</key>
    <array>
        <dict>
            <key>Script</key>
            <string>replace.js</string>
        </dict>
    </array>
</dict>
```

```
// Adding a content script - Info.plist

<key>NSExtension</key>
<dict>
   <key>SFSafariContentScript</key>
   <array>
      <dict>
         <key>Script</key>
         <string>replace.js</string>
      </dict>
   </array>
</dict>
```

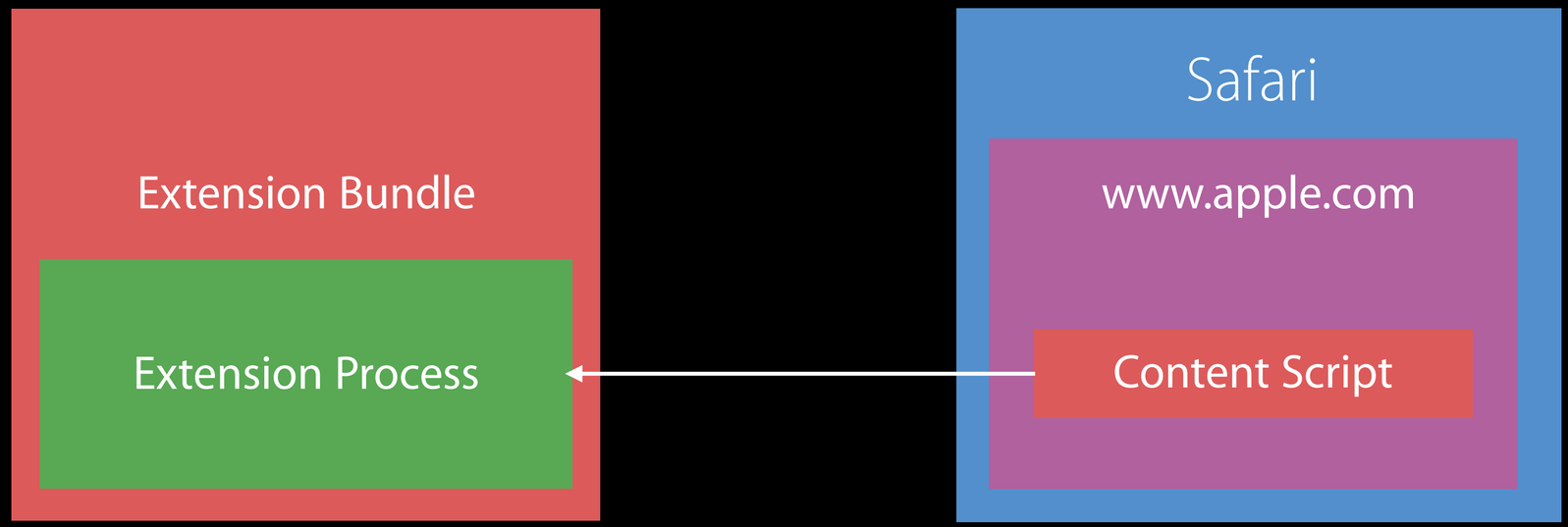
```
// Adding a content script - Info.plist

<key>NSExtension</key>
<dict>
   <key>SFSafariContentScript</key>
   <array>
      <dict>
         <key>Script</key>
         <string>replace.js</string>
      </dict>
   </array>
</dict>
```

Page Modification

Content Script Architecture

Page Modification

Content Script Architecture




```
safari.extension.dispatchMessage(“GetWordsAndReplacements");
```

```
// Receiving a message from a content script

import SafariServices

class SafariExtensionHandler: SFSafariExtensionHandler {
   override func messageReceived(withName messageName: String,
      from page: SFSafariPage, userInfo: [NSObject : AnyObject]!) {
      if (messageName == "GetWordsAndReplacements") {
         page.dispatchMessageToScript(withName: "WordsAndReplacements",
            userInfo: ["Bear": "🐻"]);
         }
      }
   }
```

```
// Receiving a message from a content script

import SafariServices

class SafariExtensionHandler: SFSafariExtensionHandler {
    override func messageReceived(withName messageName: String,
        from page: SFSafariPage, userInfo: [NSObject : AnyObject]!) {
        if (messageName == "GetWordsAndReplacements") {
            page.dispatchMessageToScript(withName: "WordsAndReplacements",
                userInfo: ["Bear": "🐻"]);
        }
    }
}
```

```
// Receiving a message from a content script

import SafariServices

class SafariExtensionHandler: SFSafariExtensionHandler {
   override func messageReceived(withName messageName: String,
      from page: SFSafariPage, userInfo: [NSObject : AnyObject]!) {
      if (messageName == "GetWordsAndReplacements") {
         page.dispatchMessageToScript(withName: "WordsAndReplacements",
            userInfo: ["Bear": "🐻"]);
         }
      }
   }
```

```
// Receiving a message from a content script

import SafariServices

class SafariExtensionHandler: SFSafariExtensionHandler {
   override func messageReceived(withName messageName: String,
      from page: SFSafariPage, userInfo: [NSObject : AnyObject]!) {
      if (messageName == "GetWordsAndReplacements") {
         page.dispatchMessageToScript(withName: "WordsAndReplacements",
            userInfo: ["Bear": "🐻"]);
      }
   }
}
```

```
// Sending a message back to the content script

import SafariServices

class SafariExtensionHandler: SFSafariExtensionHandler {
   override func messageReceived(withName messageName: String,
      from page: SFSafariPage, userInfo: [NSObject : AnyObject]!) {
      if (messageName == "GetWordsAndReplacements") {
         page.dispatchMessageToScript(withName: "WordsAndReplacements",
            userInfo: ["Bear": "🐻"]);
         }
      }
   }
```

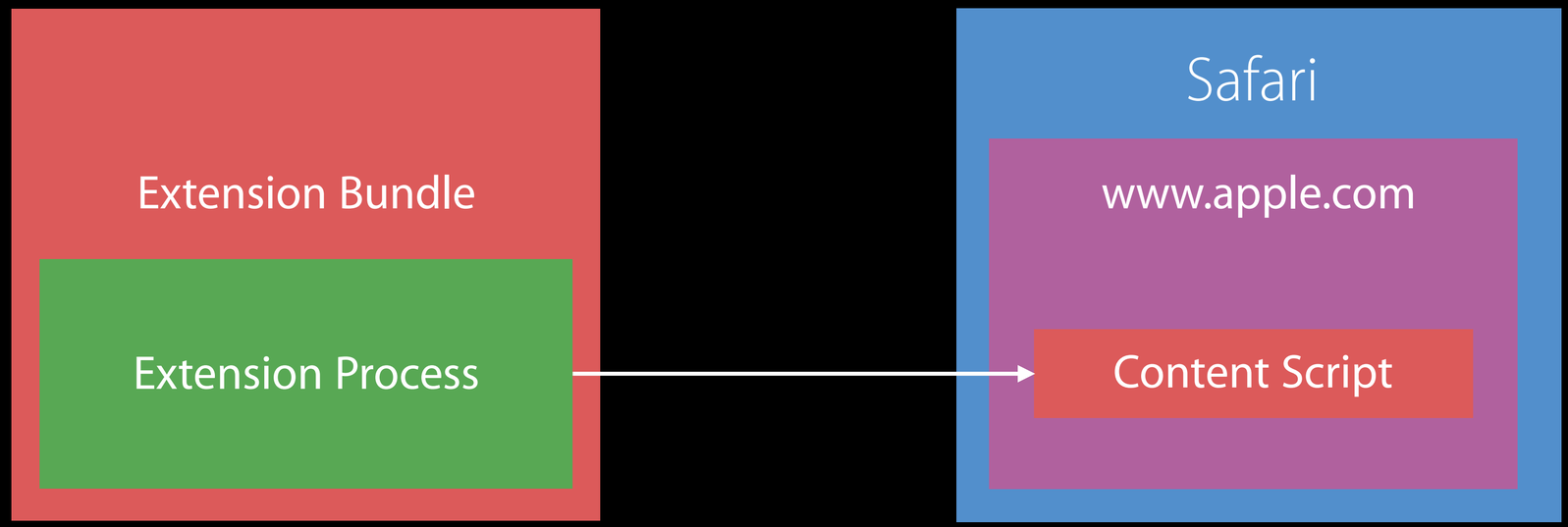
```
// Sending a message back to the content script

import SafariServices

class SafariExtensionHandler: SFSafariExtensionHandler {
   override func messageReceived(withName messageName: String,
      from page: SFSafariPage, userInfo: [NSObject : AnyObject]!) {
      if (messageName == "GetWordsAndReplacements") {
         page.dispatchMessageToScript(withName: "WordsAndReplacements",
            userInfo: ["Bear": "🐻"]);
      }
   }
}
```

Page Modification

Content Script Architecture

Page Modification

Content Script Architecture




```
page.dispatchMessageToScript(withName: "WordsAndReplacements", userInfo: ["Bear": "🐻"]);
```

```
// Listening for messages inside a content script

safari.self.addEventListener("message", messageHandler);

function messageHandler(event)
{
   if (event.name === "WordsAndReplacements") {
      var wordReplacementMap = event.message;
      for (var wordToReplace in wordReplacementMap) {
         replace(document.body, wordToReplace, wordReplacementMap[wordToReplace]);
      }
   }
}
```

```
// Listening for messages inside a content script

safari.self.addEventListener("message", messageHandler);

function messageHandler(event)
{
    if (event.name === "WordsAndReplacements") {
        var wordReplacementMap = event.message;
        for (var wordToReplace in wordReplacementMap) {
            replace(document.body, wordToReplace, wordReplacementMap[wordToReplace]);
        }
    }
}
```

```
// Listening for messages inside a content script

safari.self.addEventListener("message", messageHandler);

function messageHandler(event)
{
   if (event.name === "WordsAndReplacements") {
      var wordReplacementMap = event.message;
      for (var wordToReplace in wordReplacementMap) {
         replace(document.body, wordToReplace, wordReplacementMap[wordToReplace]);
      }
   }
}
```

```
// Listening for messages inside a content script

safari.self.addEventListener("message", messageHandler);

function messageHandler(event)
{
    if (event.name === "WordsAndReplacements") {
        var wordReplacementMap = event.message;
        for (var wordToReplace in wordReplacementMap) {
            replace(document.body, wordToReplace, wordReplacementMap[wordToReplace]);
        }
    }
}
```

```
// Listening for messages inside a content script

safari.self.addEventListener("message", messageHandler);

function messageHandler(event)
{
   if (event.name === "WordsAndReplacements") {
      var wordReplacementMap = event.message;
      for (var wordToReplace in wordReplacementMap) {
         replace(document.body, wordToReplace, wordReplacementMap[wordToReplace]);
      }
   }
}
```

```
// Listening for messages inside a content script

safari.self.addEventListener("message", messageHandler);

function messageHandler(event)
{
   if (event.name === "WordsAndReplacements") {
      var wordReplacementMap = event.message;
      for (var wordToReplace in wordReplacementMap) {
         replace(document.body, wordToReplace, wordReplacementMap[wordToReplace]);
      }
   }
}
```

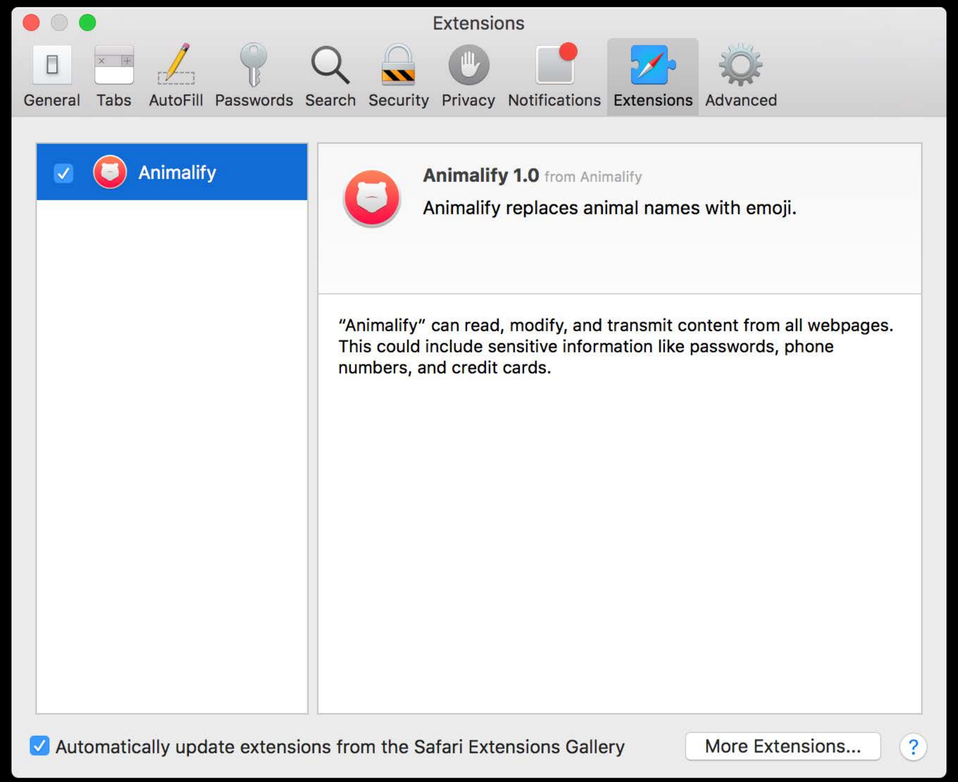
Safari App Extensions

Website Access Level

Safari App Extensions

Website Access Level

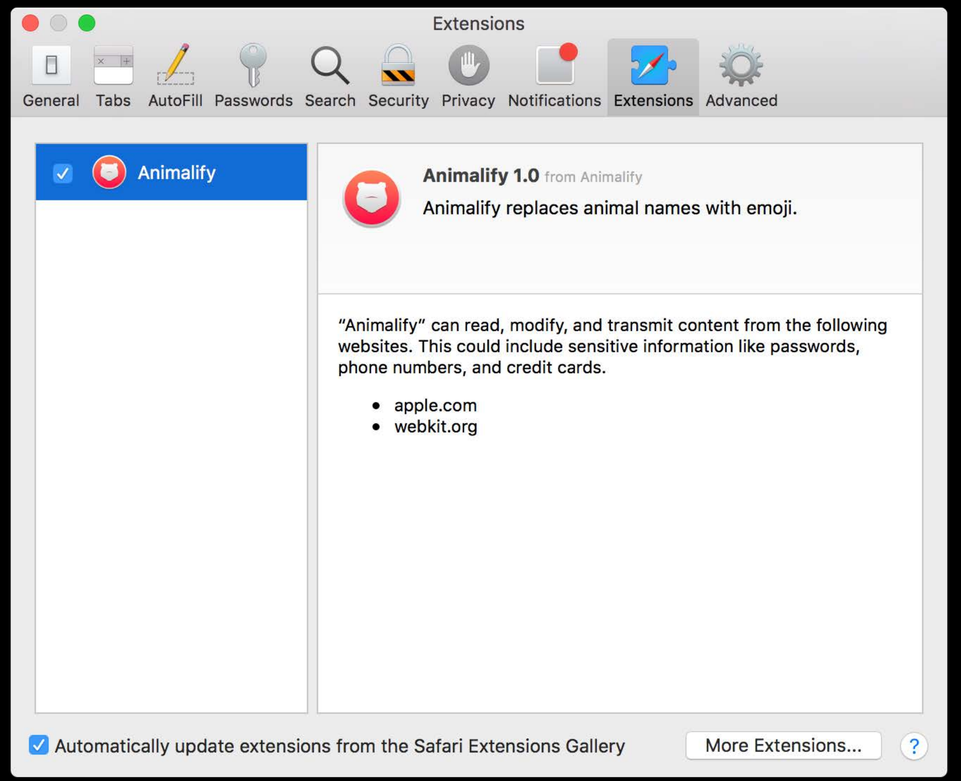
```
<key>NSExtension</key>
<dict>
    <key>SFSafariWebsiteAccess</key>
    <dict>
        <key>Level</key>
        <string>All</string>
    </dict>
</dict>
```



Safari App Extensions

Website Access Level

```
<key>NSExtension</key>
<dict>
    <key>SFSafariWebsiteAccess</key>
    <dict>
        <key>Level</key>
        <string>Some</string>
        <key>Allowed Domains</key>
        <array>
            <string>*.apple.com</string>
            <string>*.webkit.org</string>
        </array>
    </dict>
</dict>
```



Demo

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Extending Safari's UI

Damian Kaleta Safari Engineer

Notebook

Notes:

2016-06-07 19:50:31 +0000
Hello WWDC

2016-06-07 19:51:08 +0000
Session 214

2016-06-07 19:51:45 +0000
Safari Technology Preview Release 5 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 5 of Safari Technology Preview covers WebKit revisions 200418–201083.

Delete Last Note | Delete All Notes

Insert New Note

Note | Add

Release Notes for Safari Technology Preview 6

Jun 8, 2016 by Timothy Hatcher @xeenon

Safari Technology Preview Release 6 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 6 of Safari Technology Preview covers WebKit revisions 201084–201541.

JavaScript

- Added support for trailing commas in function parameters per draft ECMAScript spec (r201488)
- Improved `RegExp` matching when the result array becomes large (r201451)
- Made `RegExp` throw an exception instead of a crash when matching deeply nested subexpressions (r201412)
- Made `TypedArray.prototype.slice` no longer throw an exception if no arguments are provided (r201364)
- Improved performance of `TypedArray` access by 30% in the 64-bit low-level interpreter (r201335)

Release Notes for Safari Technology Preview 6

Jun 8, 2016 by Timothy Hatcher @xeenon

Safari Technology Preview Release 6 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 6 of Safari Technology Preview 01541.

JavaScript

- Added support for trailing commas in f AScript spec (r201488)
- Improved `RegExp` matching when the result array becomes large (r201451)
- Made `RegExp` throw an exception instead of a crash when matching deeply nested subexpressions (r201412)
- Made `TypedArray.prototype.slice` no longer throw an exception if no arguments are provided (r201364)
- Improved performance of `TypedArray` access by 30% in the 64-bit low-level interpreter (r201335)

Release Notes for Safari Technology Preview 6

Jun 8, 2016 by Timothy Hatcher @xeenon

Safari Technology Preview Release 6 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 6 of Safari Technology Preview 01541.

JavaScript

- Added support for trailing commas in f AScript spec (r201488)
- Improved `RegExp` matching when the result array becomes large (r201451)
- Made `RegExp` throw an exception instead of a crash when matching deeply nested subexpressions (r201412)
- Made `TypedArray.prototype.slice` no longer throw an exception if no arguments are provided (r201364)
- Improved performance of `TypedArray` access by 30% in the 64-bit low-level interpreter (r201335)

webkit.org

Your latest note:

Safari Technology Preview Release 6 is now available for download.

Modify

Web
Open Sou

Blog Downloads Feature Status Reporting Bugs Contribute

Re afari Technology Preview 6

Jun 8, 2016 by Timothy Hatcher @xeenon

Safari Technology Preview Release 6 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 6 of Safari Technology Preview covers WebKit revisions 201084–201541.

JavaScript

- Added support for trailing commas in function parameters per draft ECMAScript spec (r201488)
- Improved `RegExp` matching when the result array becomes large (r201451)
- Made `RegExp` throw an exception instead of a crash when matching deeply nested subexpressions (r201412)
- Made `TypedArray.prototype.slice` no longer throw an exception if no arguments are provided (r201364)
- Improved performance of `TypedArray` access by 30% in the 64-bit low-level interpreter (r201335)

Toolbar Button

Release Notes for Safari Technology Preview 6

Jun 8, 2016 by Timothy Hatcher @xeenon

Safari Technology Preview Release 6 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 6 of Safari Technology Preview covers WebKit revisions 201084–201541.

JavaScript

- Added support for trailing commas in function parameters per draft ECMAScript spec (r201488)
- Improved `RegExp` matching when the result array becomes large (r201451)
- Made `RegExp` throw an exception instead of a crash when matching deeply nested subexpressions (r201412)
- Made `TypedArray.prototype.slice` no longer throw an exception if no arguments are provided (r201364)
- Improved performance of `TypedArray` access by 30% in the 64-bit low-level interpreter (r201335)

```
// Definition
// Toolbar Button

<key>NSExtension</key>
<dict>
   <key>SFSafariToolbarItem</key>
   <dict>
      <key>Identifier</key>
      <string>NotebookToolbarItem</string>
      <key>Label</key>
      <string>Show Snippets</string>
      <key>Image</key>
      <string>ToolbarItemIcon.pdf</string>
      <key>Action</key>
      <string>Command</string>
   </dict>
</dict>
```

```
// Definition
// Toolbar Button

<key>NSExtension</key>
<dict>
   <key>SFSafariToolbarItem</key>
   <dict>
      <key>Identifier</key>
      <string>NotebookToolbarItem</string>
      <key>Label</key>
      <string>Show Snippets</string>
      <key>Image</key>
      <string>ToolbarItemIcon.pdf</string>
      <key>Action</key>
      <string>Command</string>
   </dict>
</dict>
```

```
    // Definition
    // Toolbar Button

    <key>NSExtension</key>
    <dict>
        <key>SFSafariToolbarItem</key>
        <dict>
            <key>Identifier</key>
            <string>NotebookToolbarItem</string>
            <key>Label</key>
            <string>Show Snippets</string>
            <key>Image</key>
            <string>ToolbarItemIcon.pdf</string>
            <key>Action</key>
            <string>Command</string>
        </dict>
    </dict>
```

Toolbar Button

APIs

All methods should be defined on SFSafariExtensionHandler

```
optional public func toolbarItemClicked(in window: SFSafariWindow)
```

```
optional public func validateToolbarItem(in window: SFSafariWindow,
    validationHandler: (enabled: Bool, badgeText: String) -> Swift.Void)
```

webkit.org

Your latest note:

Safari Technology Preview Release 6 is now available for download.

Modify

Web
Open Sou

Blog Downloads Feature Status Reporting Bugs Contribute

Re afari Technology Preview 6

Jun 8, 2016 by Timothy Hatcher @xeenon

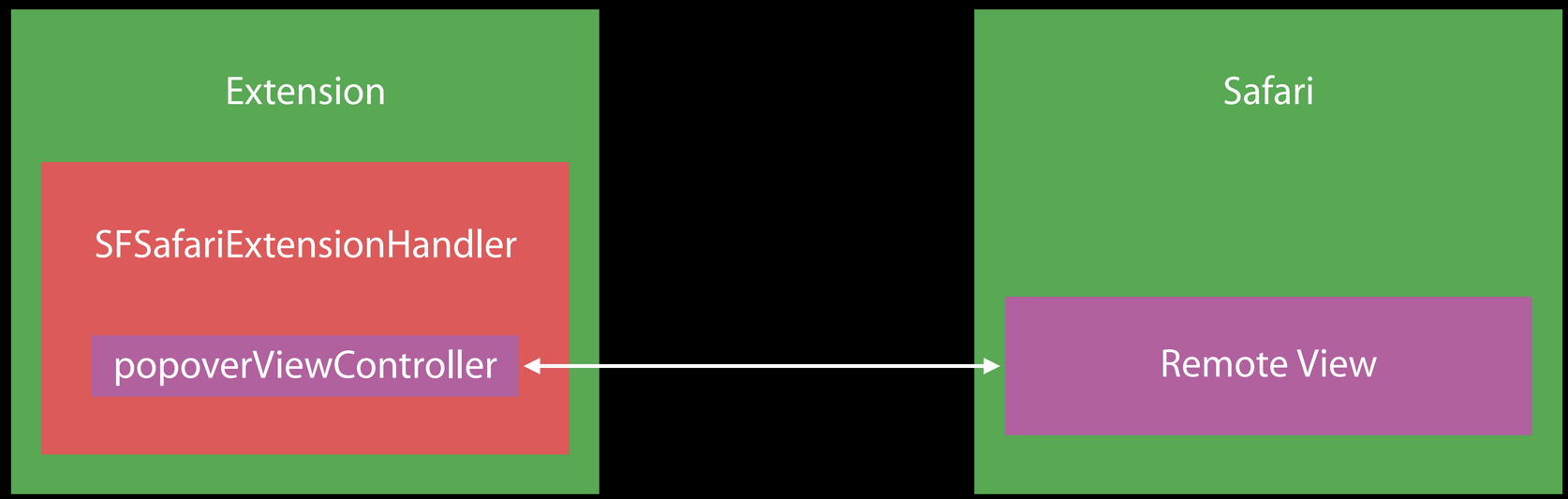
Safari Technology Preview Release 6 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 6 of Safari Technology Preview covers WebKit revisions 201084–201541.

JavaScript

- Added support for trailing commas in function parameters per draft ECMAScript spec (r201488)
- Improved `RegExp` matching when the result array becomes large (r201451)
- Made `RegExp` throw an exception instead of a crash when matching deeply nested subexpressions (r201412)
- Made `TypedArray.prototype.slice` no longer throw an exception if no arguments are provided (r201364)
- Improved performance of `TypedArray` access by 30% in the 64-bit low-level interpreter (r201335)

Popover

Architecture



```
// Definition
// Popover

<key>NSExtension</key>
<dict>
   <key>SFSafariToolbarItem</key>
   <dict>
      <key>Identifier</key>
      <string>NotebookToolbarItem</string>
      <key>Label</key>
      <string>Show Snippets</string>
      <key>Image</key>
      <string>ToolbarItemIcon.pdf</string>
      <key>Action</key>
      <string>Popover</string>
   </dict>
</dict>
```

```
// Definition
// Popover

<key>NSExtension</key>
<dict>
   <key>SFSafariToolbarItem</key>
   <dict>
      <key>Identifier</key>
      <string>NotebookToolbarItem</string>
      <key>Label</key>
      <string>Show Snippets</string>
      <key>Image</key>
      <string>ToolbarItemIcon.pdf</string>
      <key>Action</key>
      <string>Popover</string>
   </dict>
</dict>
```

Popover

APIs

All methods should be defined on SFSafariExtensionHandler

```
optional public func popoverWillShow(in window: SFSafariWindow)

optional public func popoverDidClose(in window: SFSafariWindow)

optional public func popoverViewController() -> SFSafariExtensionViewController
```

Context Menu Items

Release Notes for Safari Technology Preview 6

Jun 8, 2016 by Timothy Hatcher @xeenon

Safari Technology Preview Release 6 is now available for download. If you already have Safari Technology Preview installed, you can update from the Mac App Store's Updates tab. Release 6 of Safari Technology Preview 01541.

JavaScript

- Added support for trailing commas in f AScript spec (r201488)
- Improved `RegExp` matching when the result array becomes large (r201451)
- Made `RegExp` throw an exception instead of a crash when matching deeply nested subexpressions (r201412)
- Made `TypedArray.prototype.slice` no longer throw an exception if no arguments are provided (r201364)
- Improved performance of `TypedArray` access by 30% in the 64-bit low-level interpreter (r201335)

```
// Definition
// Context Menu Items

<key>NSExtension</key>
<dict>
   <key>SFSafariContextMenu</key>
   <array>
      <dict>
         <key>Text</key>
         <string>Add Snippet to Notebook</string>
         <key>Command</key>
         <string>Add</string>
      </dict>
   </array>
</dict>
```

```
// Definition
// Context Menu Items

<key>NSExtension</key>
<dict>
   <key>SFSafariContextMenu</key>
   <array>
      <dict>
         <key>Text</key>
         <string>Add Snippet to Notebook</string>
         <key>Command</key>
         <string>Add</string>
      </dict>
   </array>
</dict>
```

```
// Definition
// Context Menu Items

<key>NSExtension</key>
<dict>
    <key>SFSafariContextMenu</key>
    <array>
        <dict>
            <key>Text</key>
            <string>Add Snippet to Notebook</string>
            <key>Command</key>
            <string>Add</string>
        </dict>
    </array>
</dict>
```

Context Menu Items

APIs

All methods are defined on SFSafariExtensionHandler

```
optional public func contextMenuItemSelected(withCommand command: String,
    in page: SFSafariPage, userInfo: [NSObject : AnyObject]? = [:])
```

Context Menu Items

Setting Context Menu User Info

```
document.addEventListener("contextmenu", handleContextMenu, false);


function handleContextMenu(event)
{
    var selectedText = window.getSelection().toString();
    safari.extension.setContextMenuEventUserInfo(event,
        { "selectedText": selectedText });
}
```

Demo

Extend Safari's UI

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Safari App Extensions

Types

Content Blockers

Page modification and communication with native code

Extending Safari's UI

Summary

Based on App Extensions

Summary

Based on App Extensions

Distributed with your Mac app

More Information

https://developer.apple.com/wwdc16/214

Related Sessions

Creating Extensions for iOS and OS X, Part 1	WWDC 2014
Creating Extensions for iOS and OS X, Part 2	WWDC 2014
App Extension Best Practices	WWDC 2015

Labs

Safari and WebKit Lab	Fort Mason	Wednesday 4:30PM
Safari and WebKit Lab	Frameworks Lab C	Friday 4:30PM

WWDC16