Native UI architecture using Web technology

- An introduction of SamuraiNative architecture -

Presented by Geek-Zoo Studio 2015 @ D2Conf Hangzhou

老郭为人民服务@weibo / gavinkwoe@github

from Geek-Zoo Studio

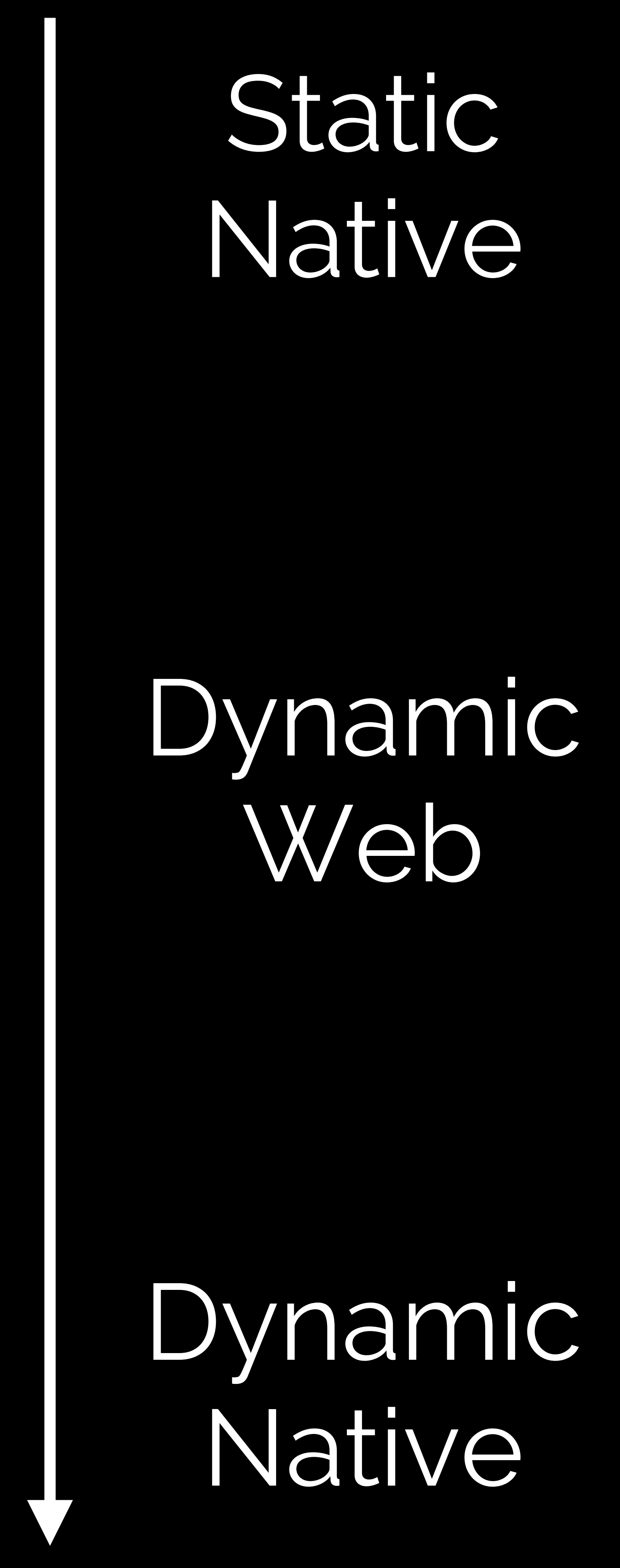
Agenda

- Review the history
- Web-Core v.s. View-Core
- Do it yourself, My-Core
- More implementation detail
- Demo

Review the history

History

- 2008 - Code or IB
- 2011 - PhoneGap
- 2013 - AutoLayout
- 2013 - BeeFramework (XML + CSS)
- 2015 - ReactNative (JSX + XML + CSS)
- 2015 - SamuraiNative (HTML + CSS)

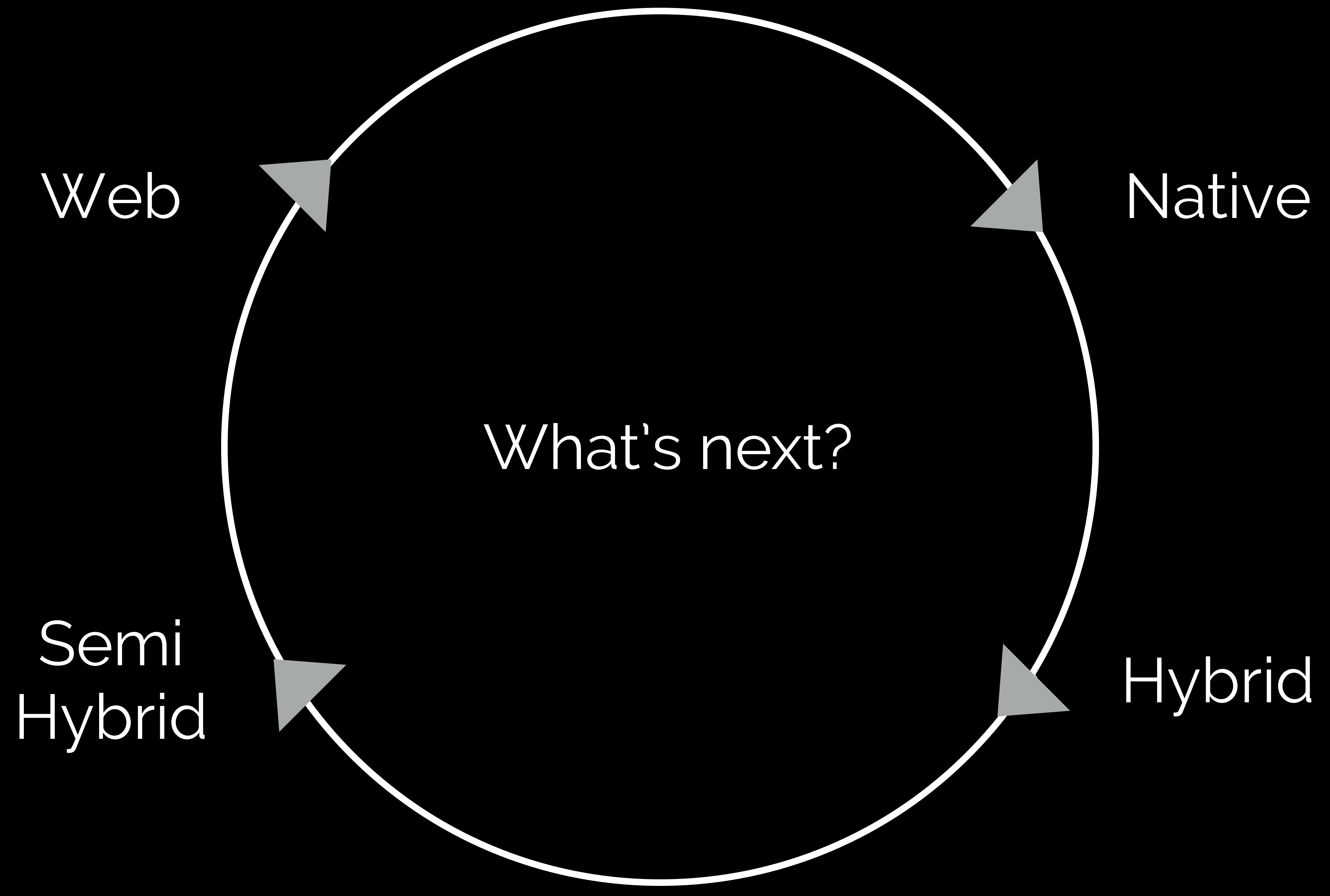


Static
Native

Dynamic
Web

Dynamic
Native

History



Web

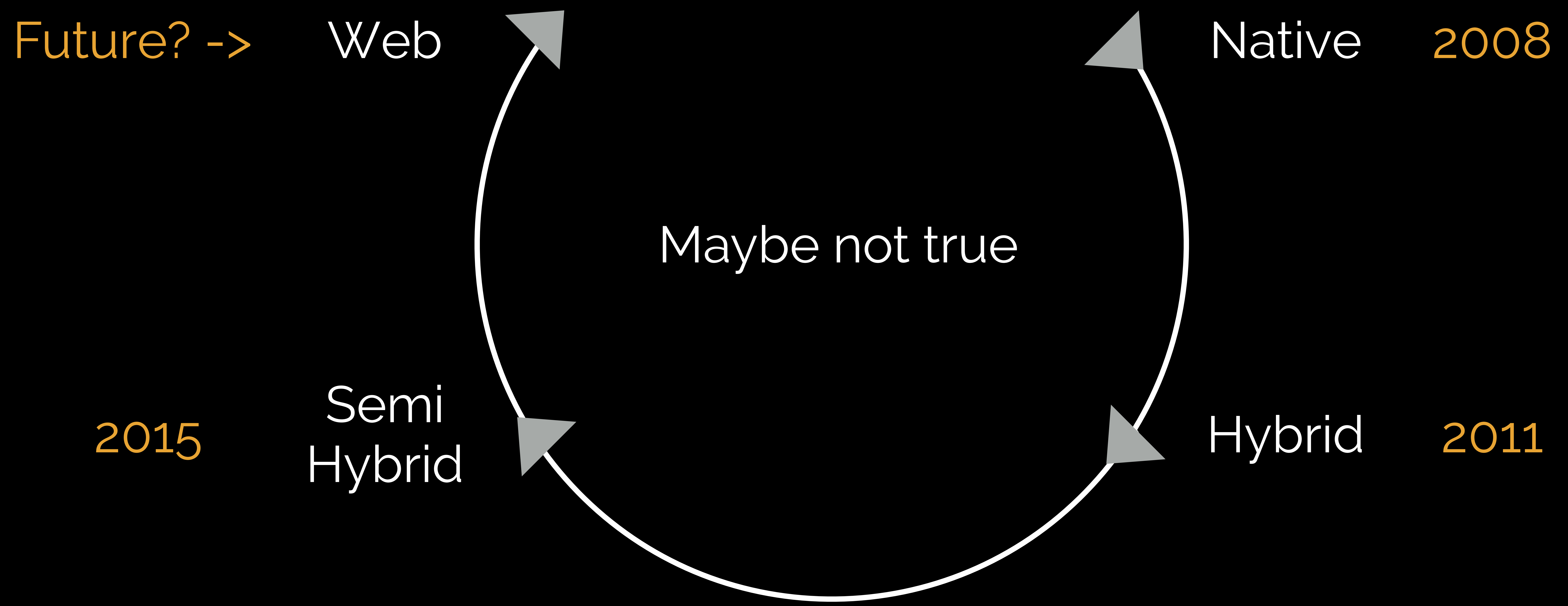
Native

What's next?

Semi Hybrid

Hybrid

History



Future? -> Web

Native 2008

Maybe not true

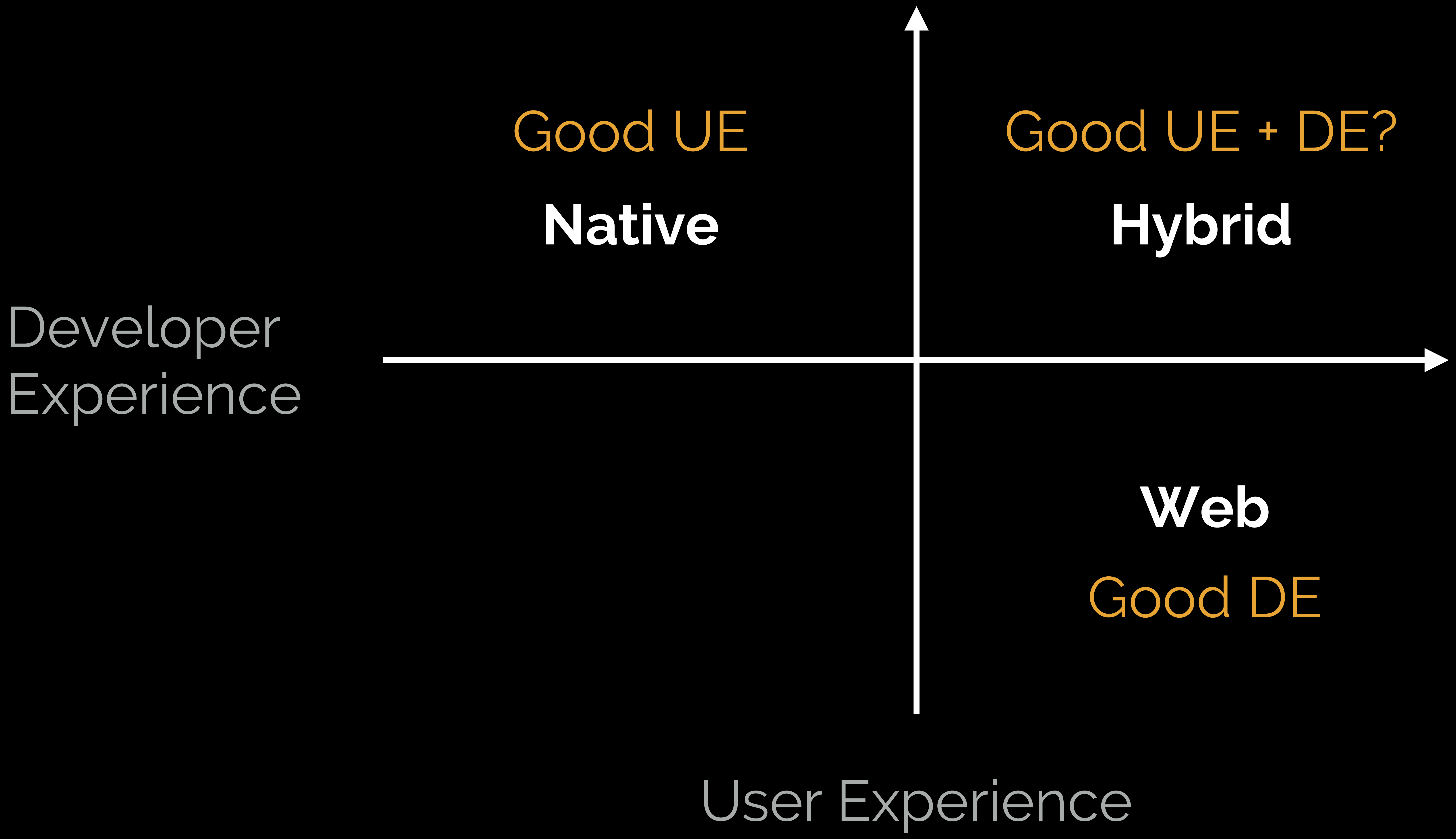
Hybrid 2011

2015 Semi Hybrid

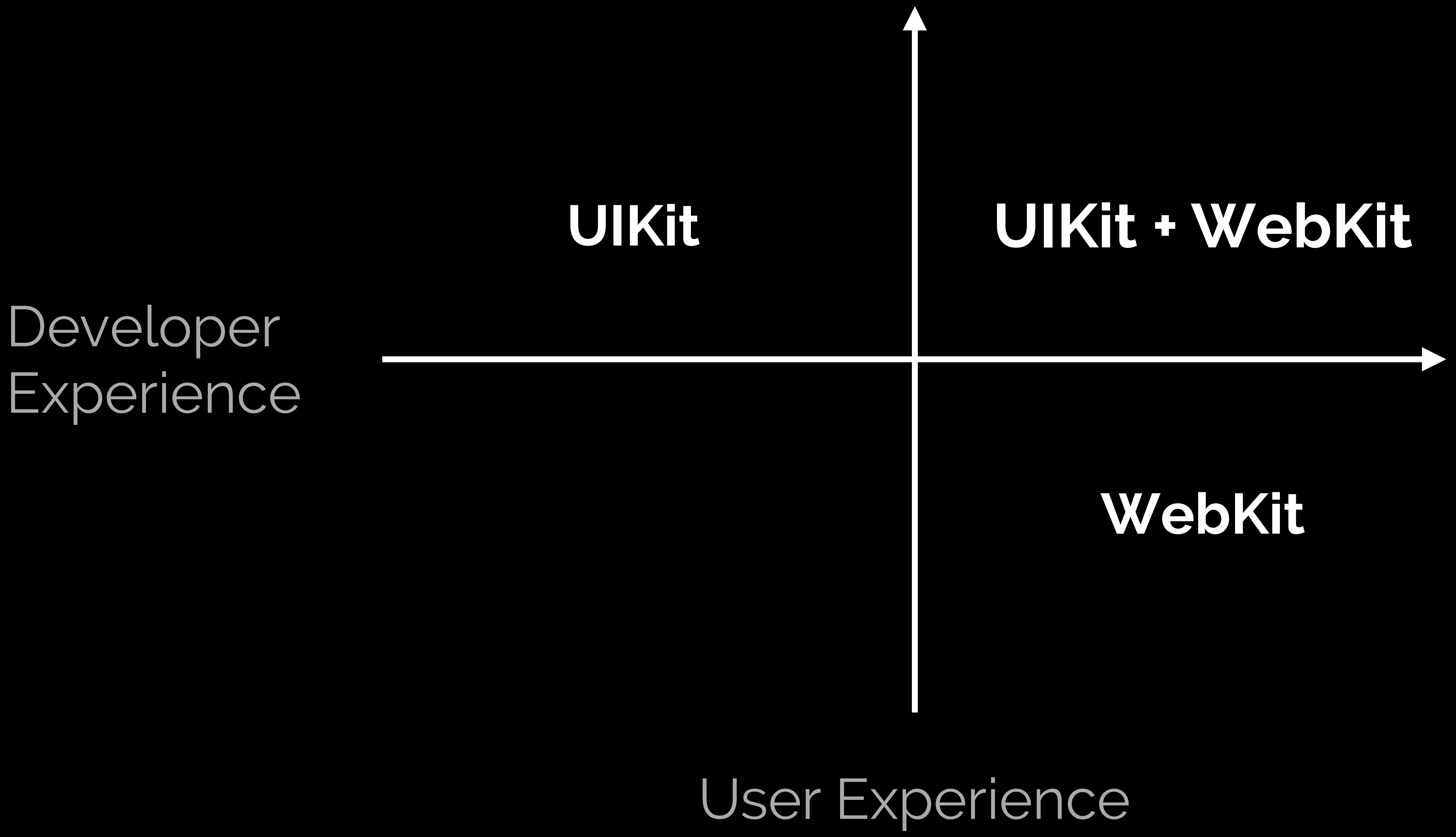
Native v.s. Web

What we are seeking for?

Native v.s. Web



Native v.s. Web


UIKit

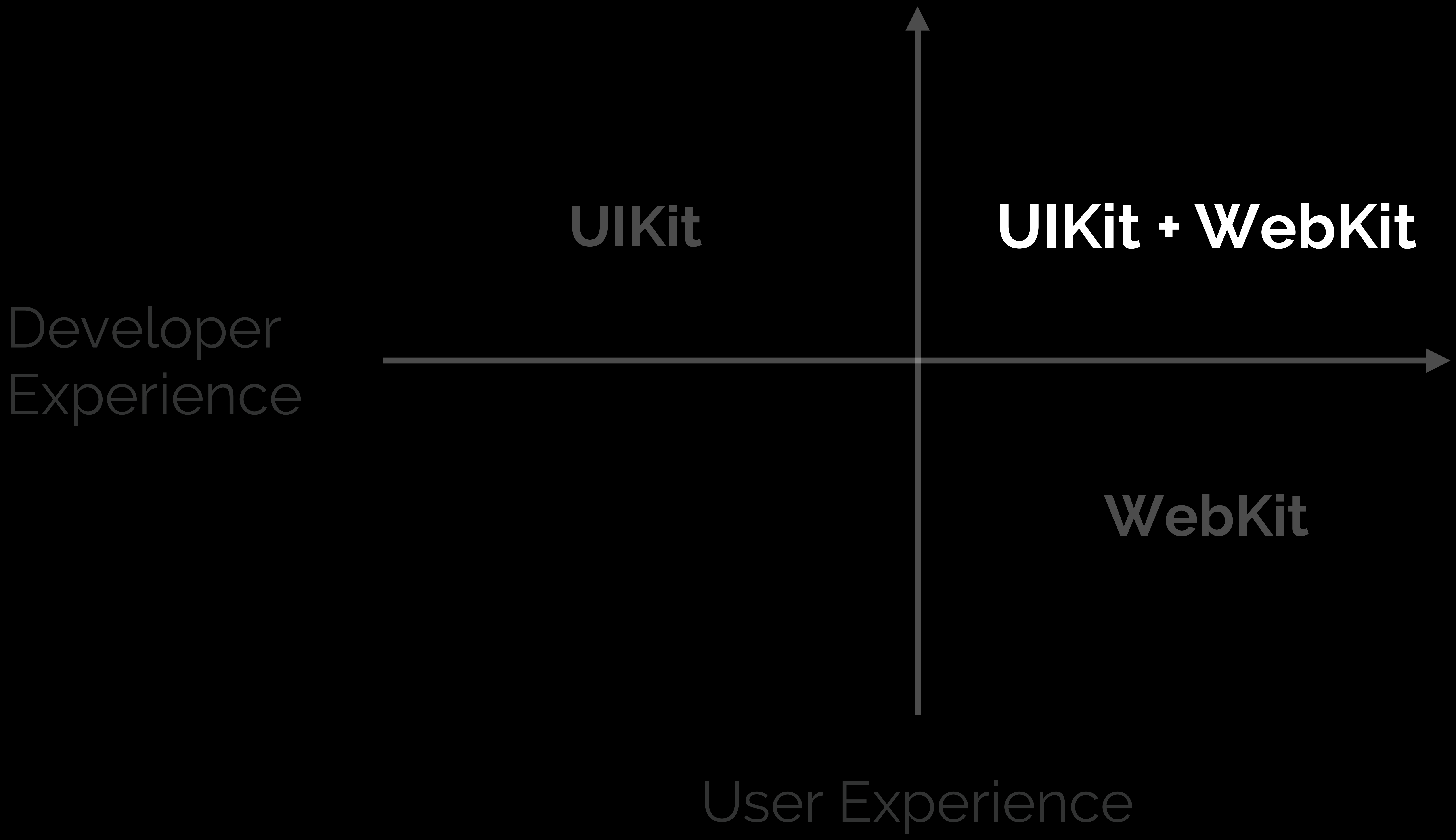
UIKit + WebKit

Developer Experience

WebKit

User Experience

Native v.s. Web



UIKit

UIKit + WebKit <- Today's topic

Developer Experience

WebKit

User Experience

Native v.s. Web

What we are seeking for?

A Dynamic Native UI solution with good DE.

Native v.s. Web

2008 - Code or IB	Code/XML -> ViewTree
2011 - PhoneGap	HTML/CSS -> RenderTree
2013 - AutoLayout	Code -> ViewTree
2013 - BeeFramework (XML + CSS)	XML/CSS -> ViewTree
2015 - ReactNative (JSX + XML + CSS)	XML/CSS -> ViewTree
2015 - SamuraiNative (HTML + CSS)	HTML/CSS -> ViewTree

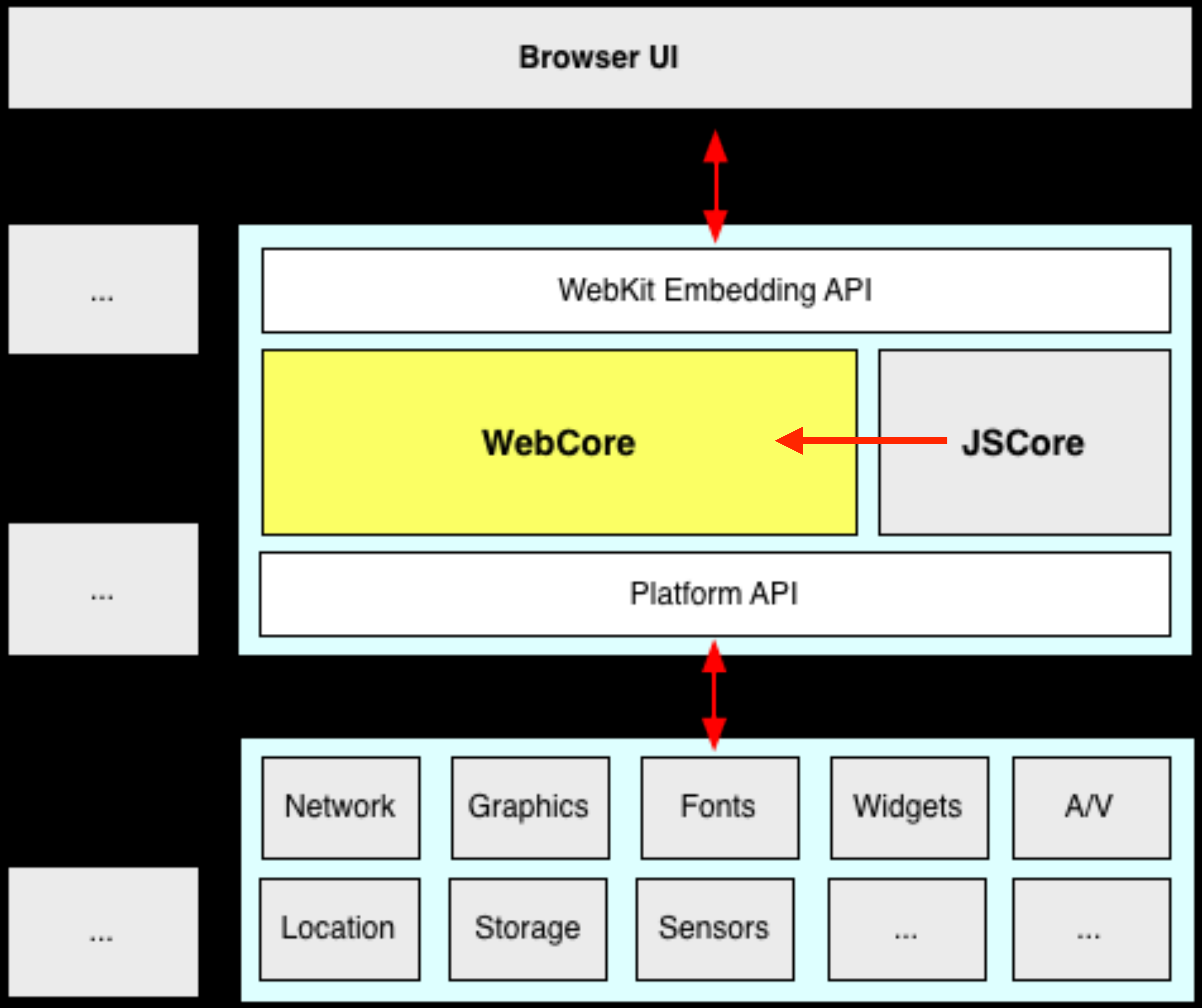
Web-Core v.s. View-Core

Traditional Web-Core

Developer ->

DOM API ->

WebCore ->

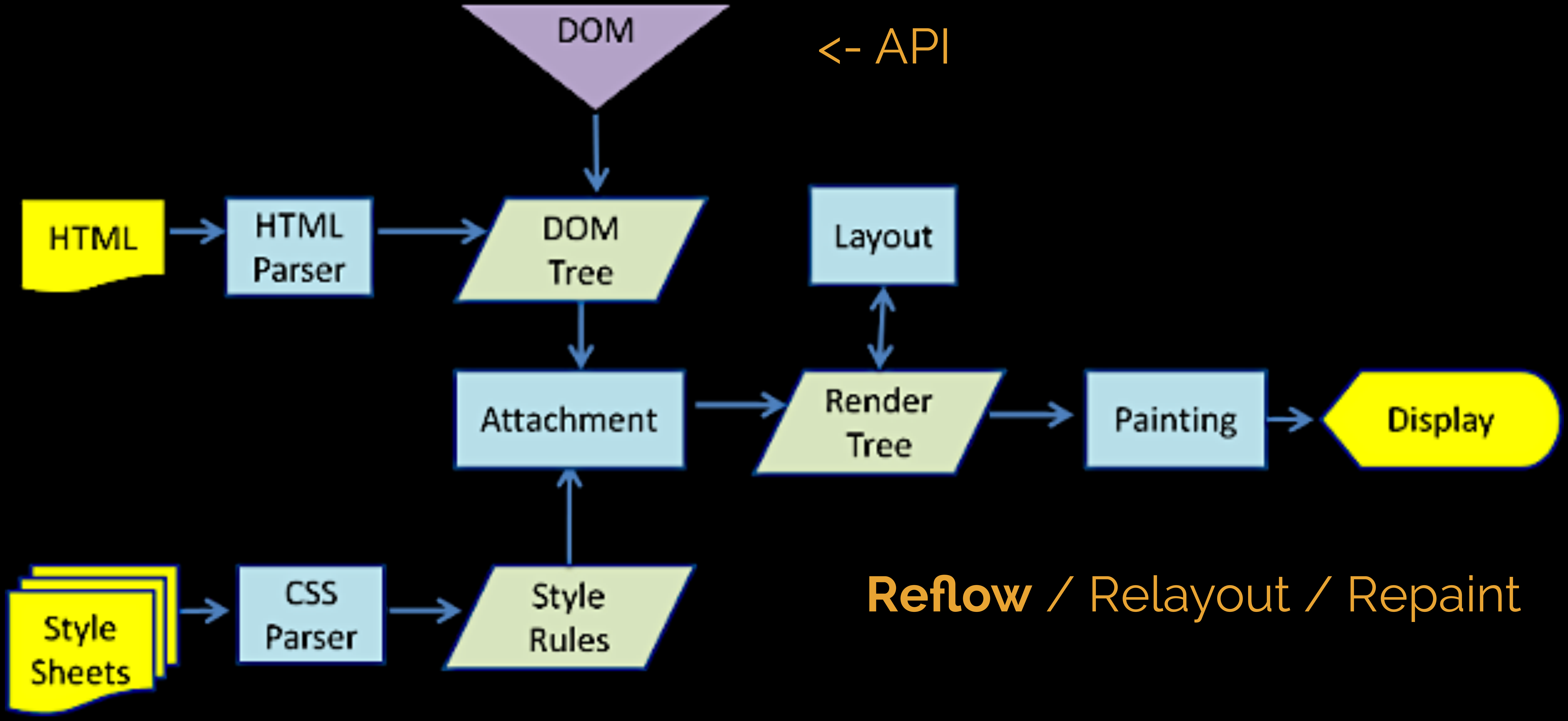


<- WebView

<- Javascript

<- JSCore

Traditional Web-Core



<- API

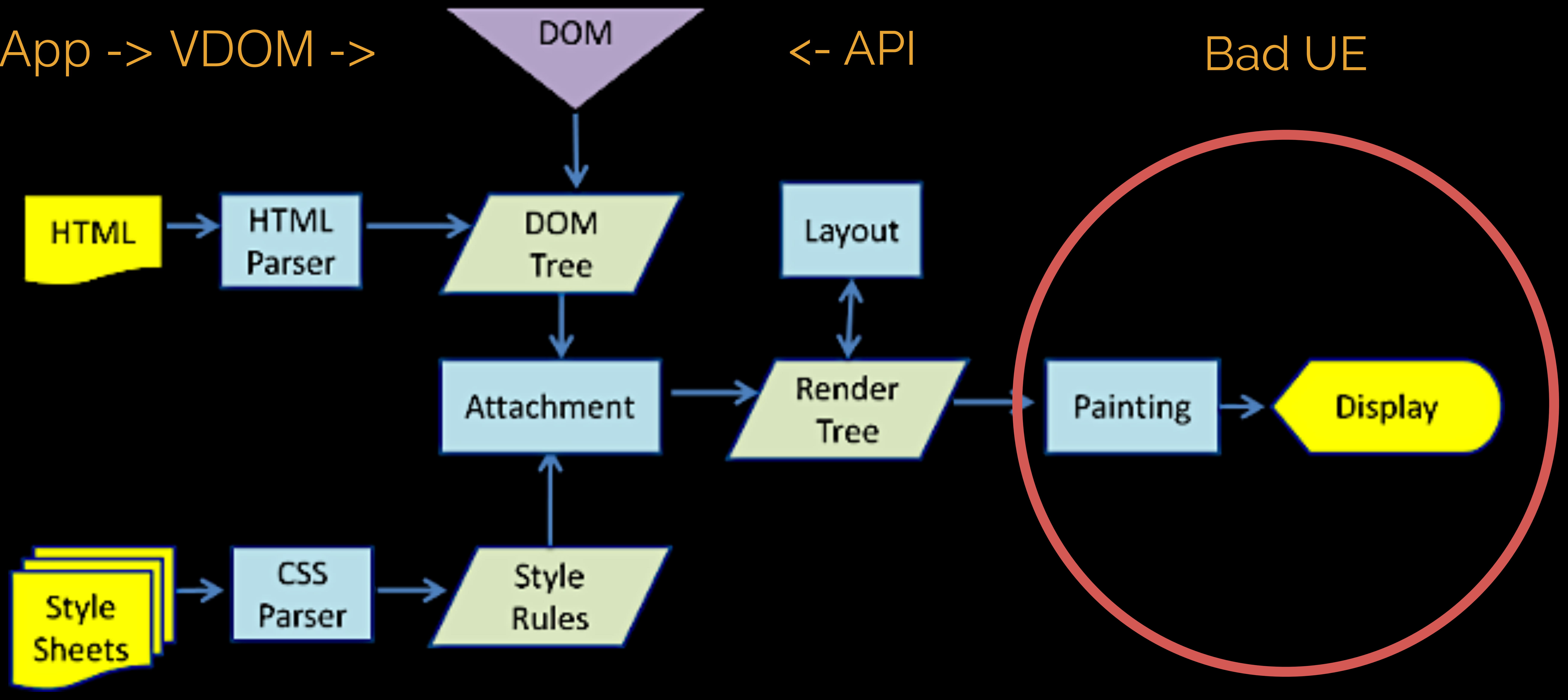
Reflow / Relayout / Repaint

Traditional Web-Core

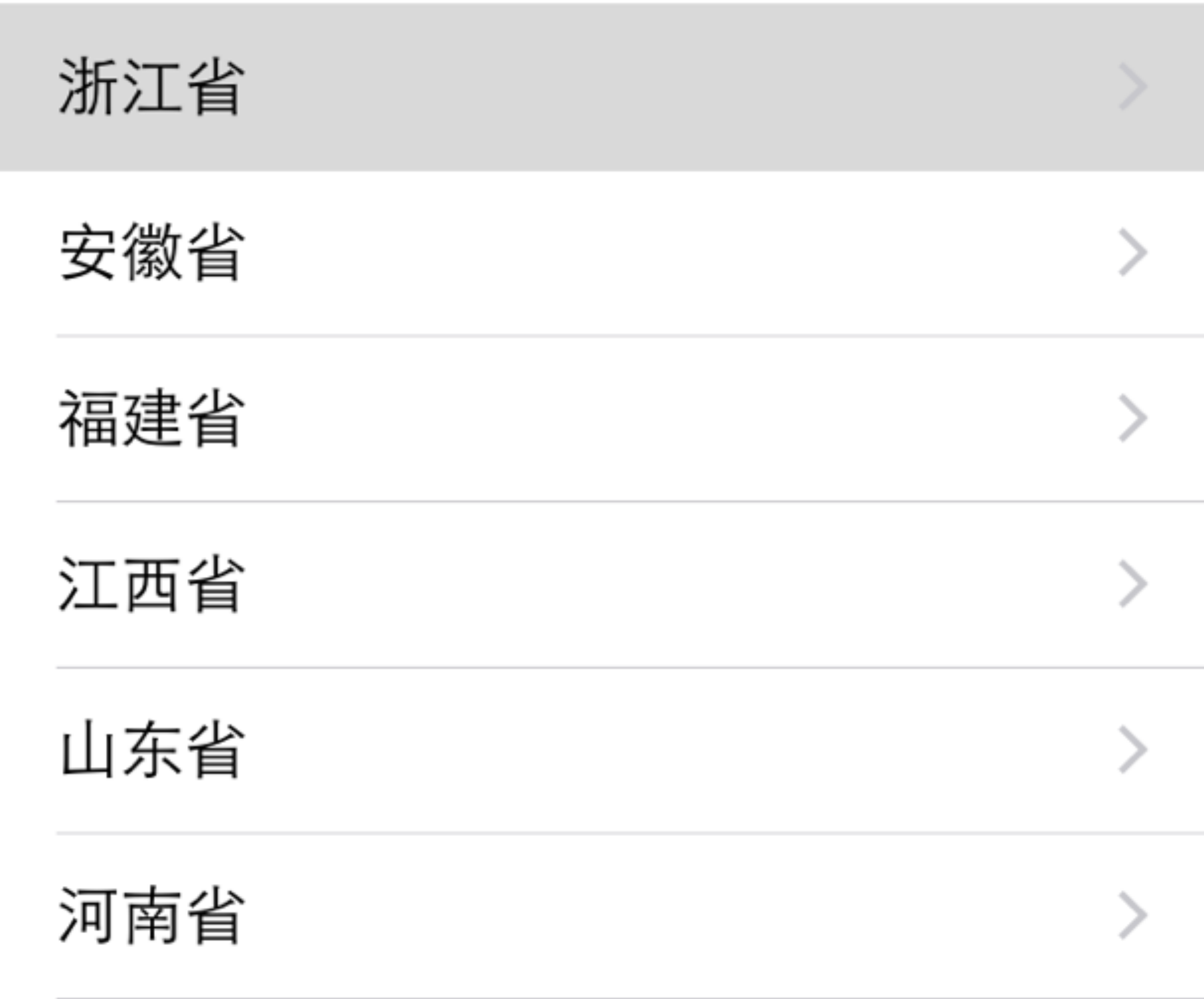
JSApp -> VDOM ->

<- API

Bad UE



Trouble in paradise



浙江省

安徽省

福建省

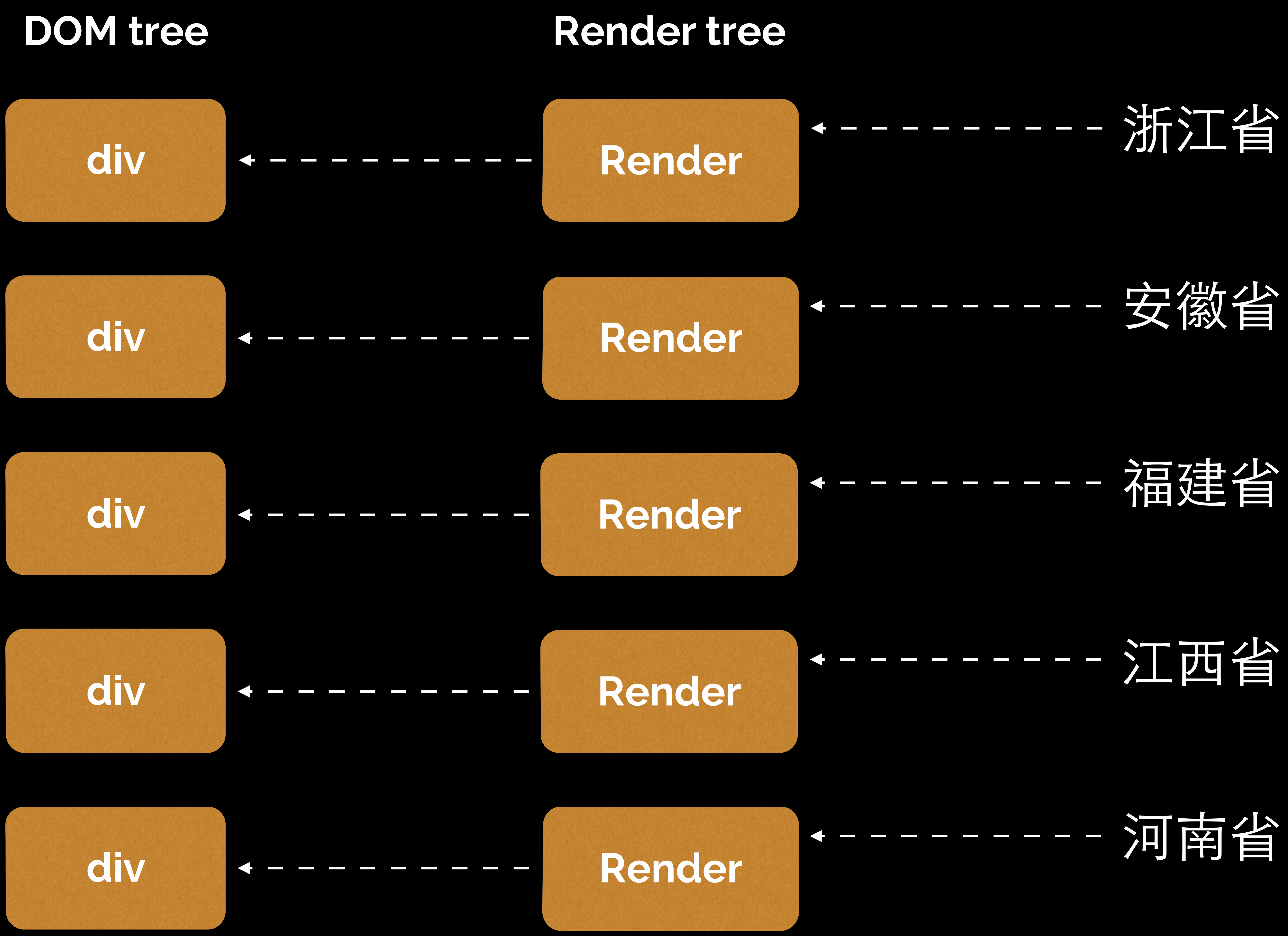
江西省

山东省

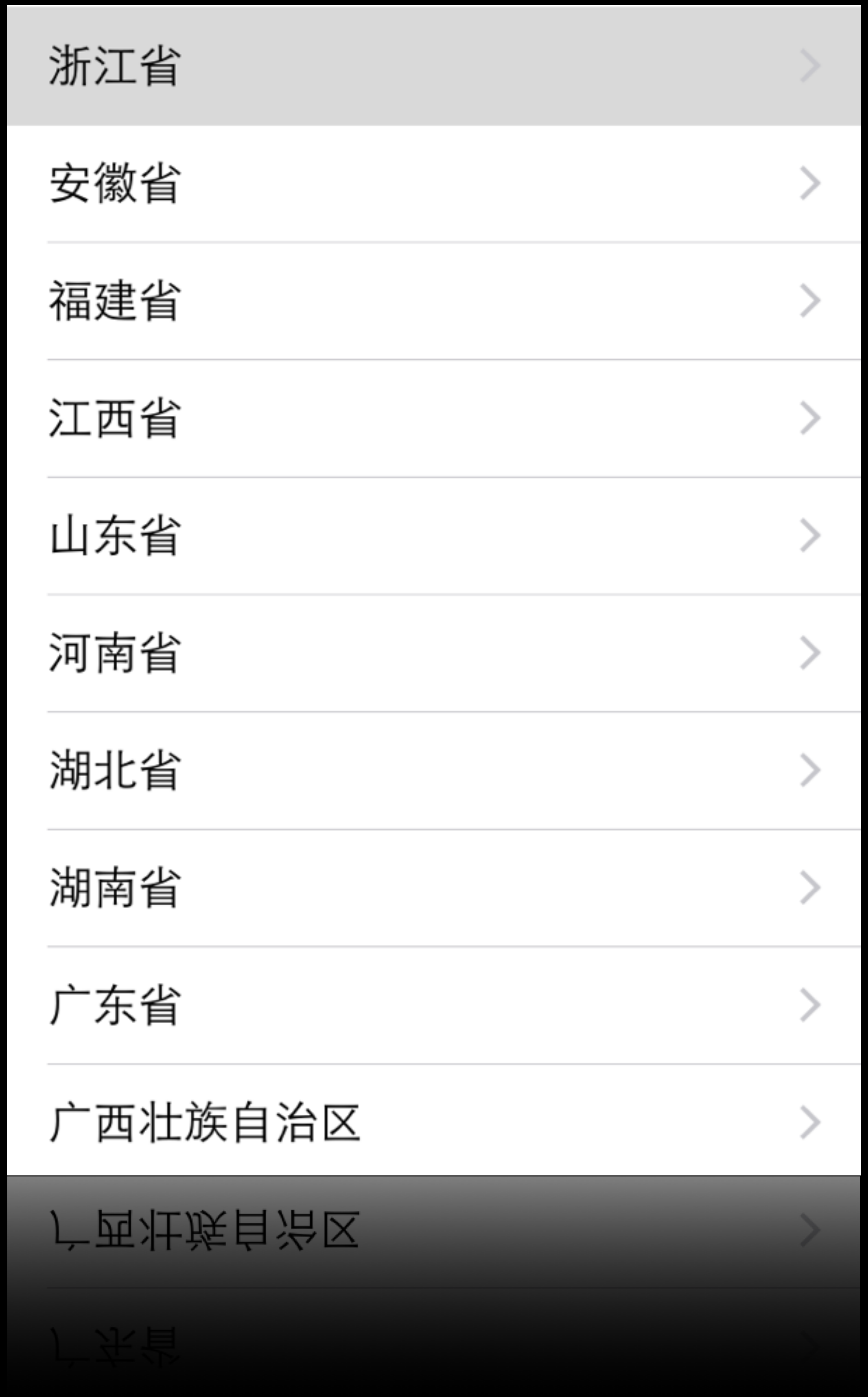
河南省

```
<div class="active">浙江省</div>
<div>安徽省</div>
<div>福建省</div>
<div>江西省</div>
<div>山东省</div>
<div>河南省</div>
```

Trouble in paradise



Trouble in paradise



浙江省

安徽省

福建省

江西省

山东省

河南省

湖北省

湖南省

广东省

广西壮族自治区

```
<div class=“active”>浙江省</div>
<div>安徽省</div>
<div>福建省</div>
<div>江西省</div>
<div>山东省</div>
<div>河南省</div>
<div>湖北省</div>
<div>广东省</div>
...
```

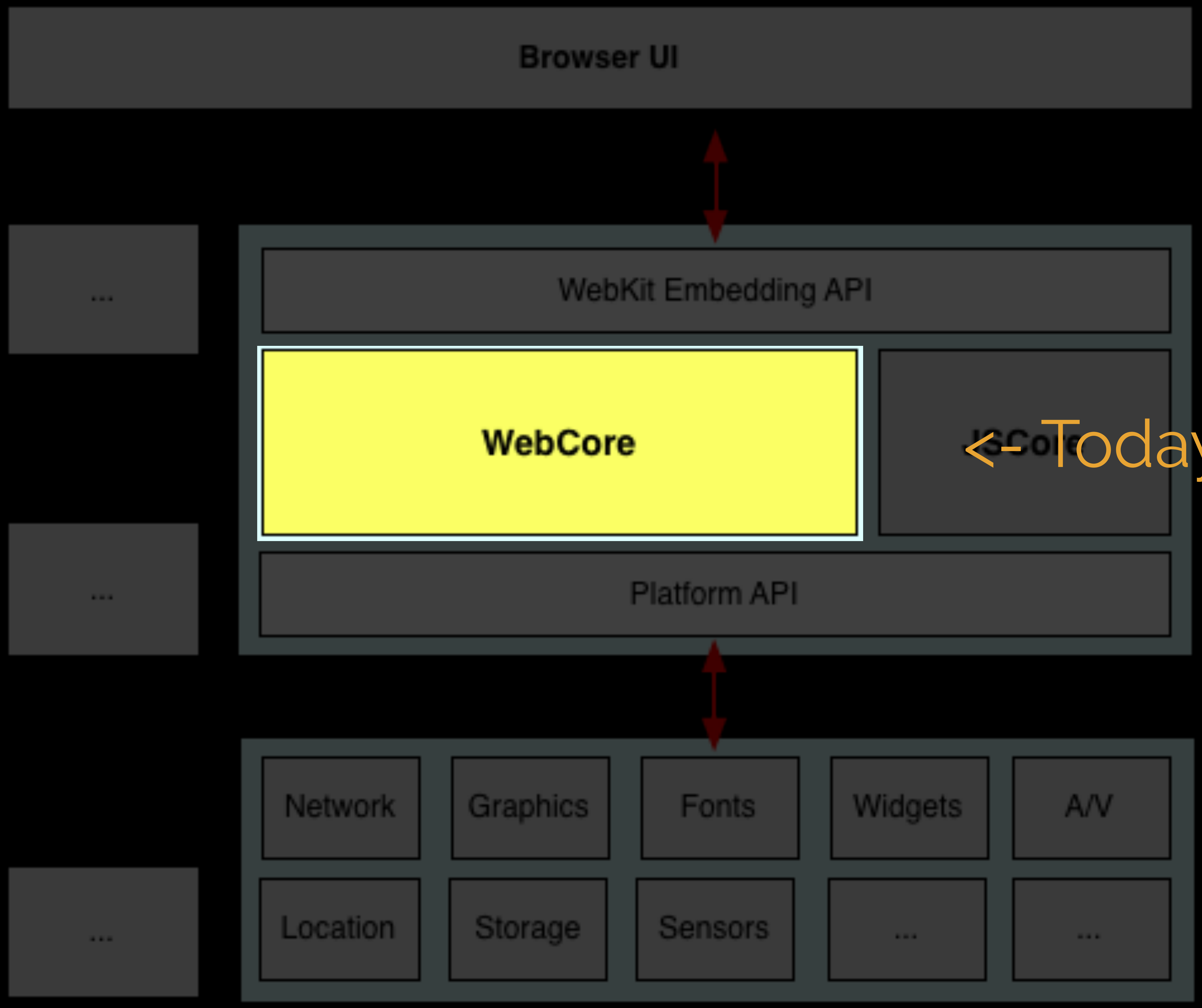
Append DOM:

1. Memory problem
2. Performance problem
3. User experience problem

Trouble in paradise

How to improve this architecture?

Trouble in paradise



Browser UI

...

WebKit Embedding API

WebCore

JSCore

...

Platform API

Network

Graphics

Fonts

Widgets

A/V

...

Location

Storage

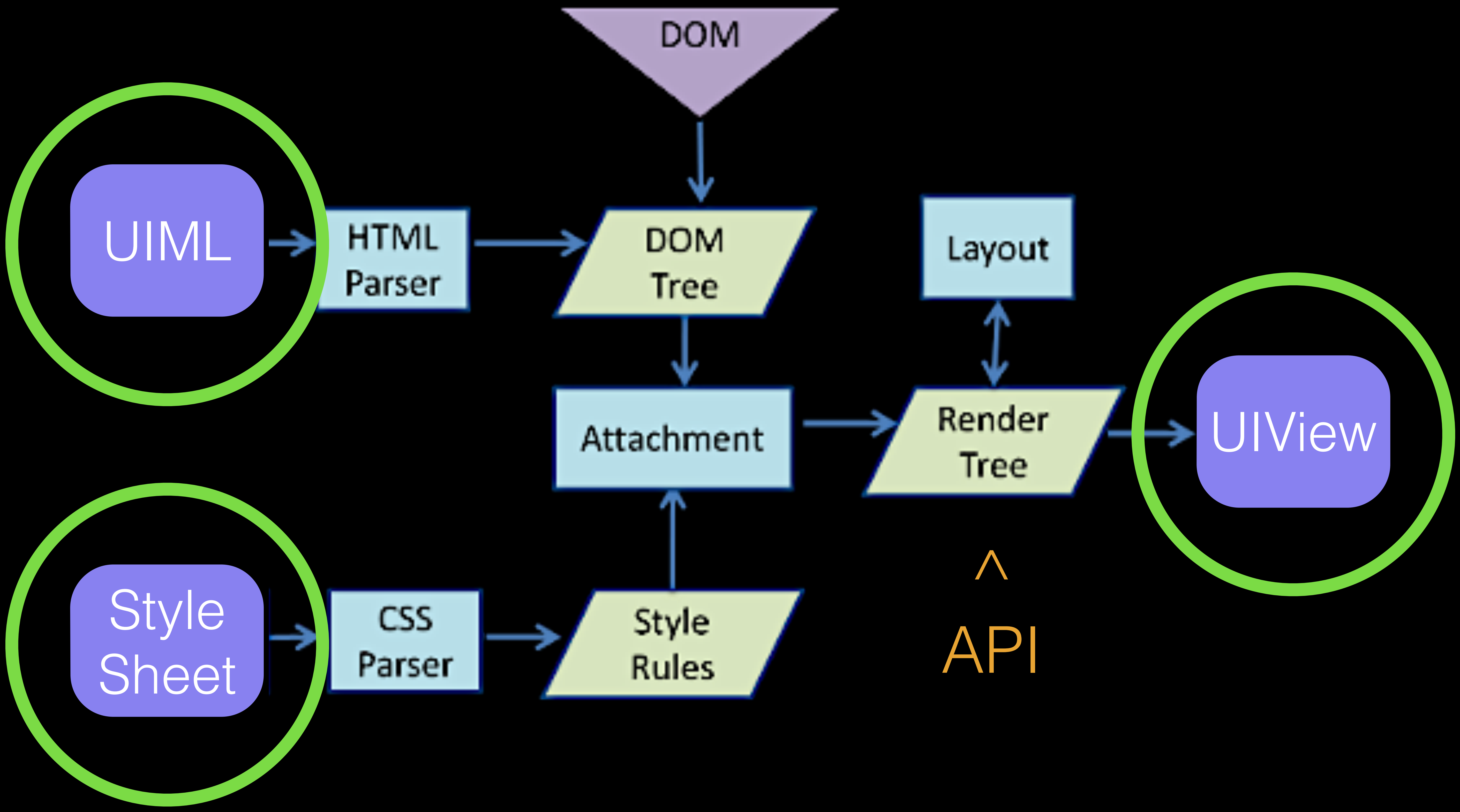
Sensors

...

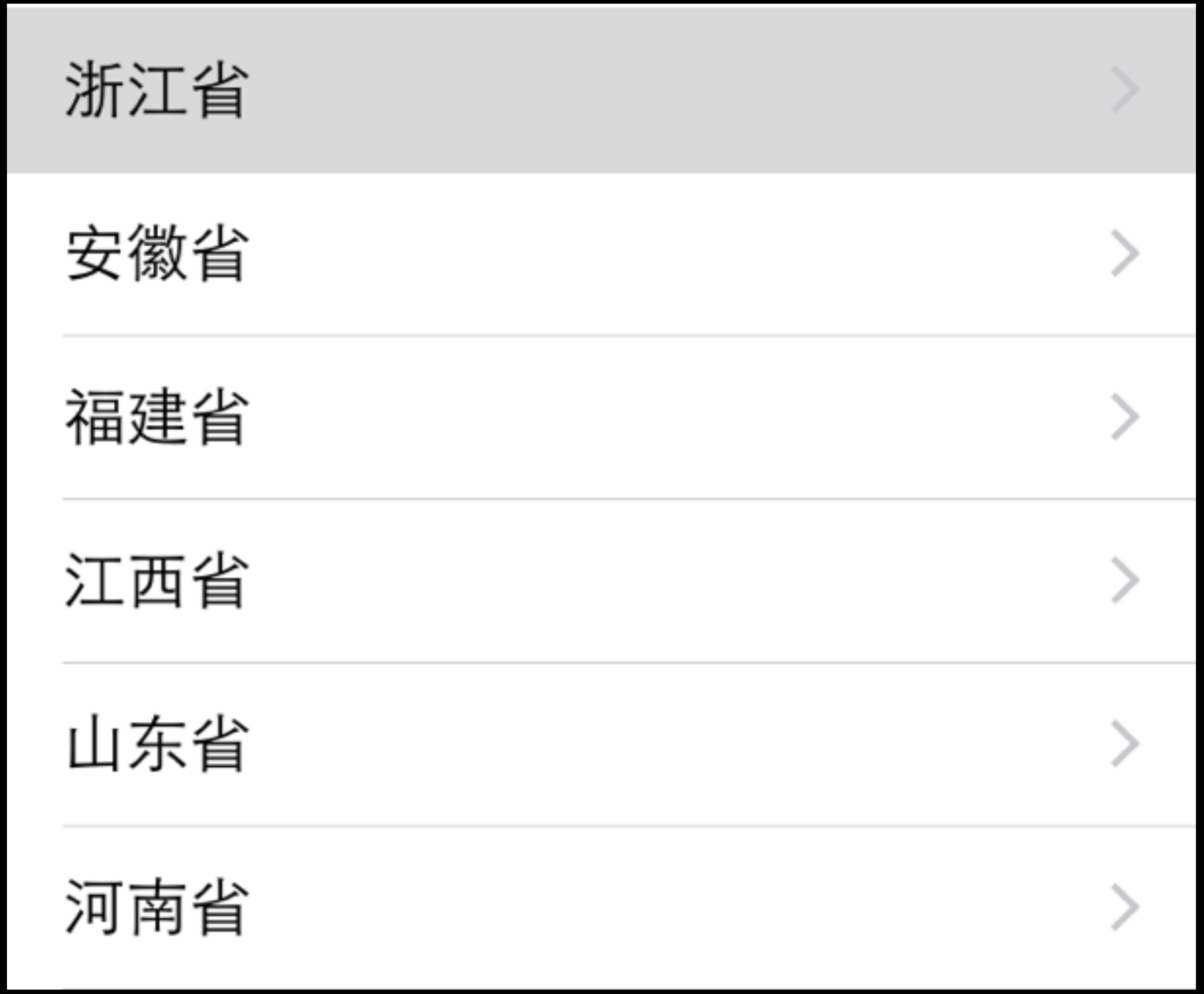
...

<- Today's topic

View-Core Solution



View-Core Solution



浙江省

安徽省

福建省

江西省

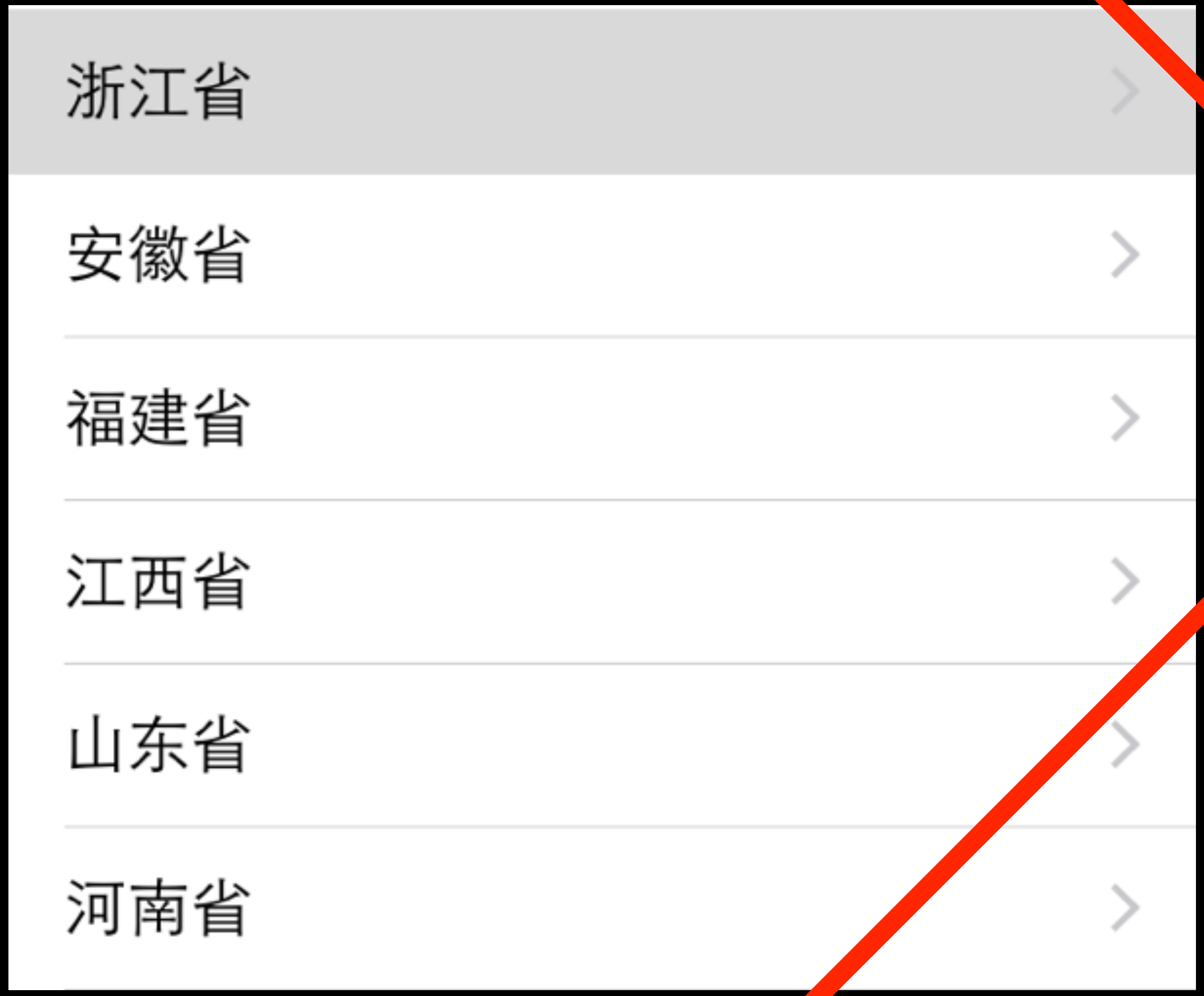
山东省

河南省

```
<UITableView>
    <UITableViewCell>浙江省</UITableViewCell>
    <UITableViewCell>安徽省</UITableViewCell>
    <UITableViewCell>福建省</UITableViewCell>
    <UITableViewCell>江西省</UITableViewCell>
    <UITableViewCell>山东省</UITableViewCell>
    <UITableViewCell>河南省</UITableViewCell>
<UITableView>
```

Simply, use <UITableViewCell/> instead of <div> ?

View-Core Solution



浙江省

安徽省

福建省

江西省

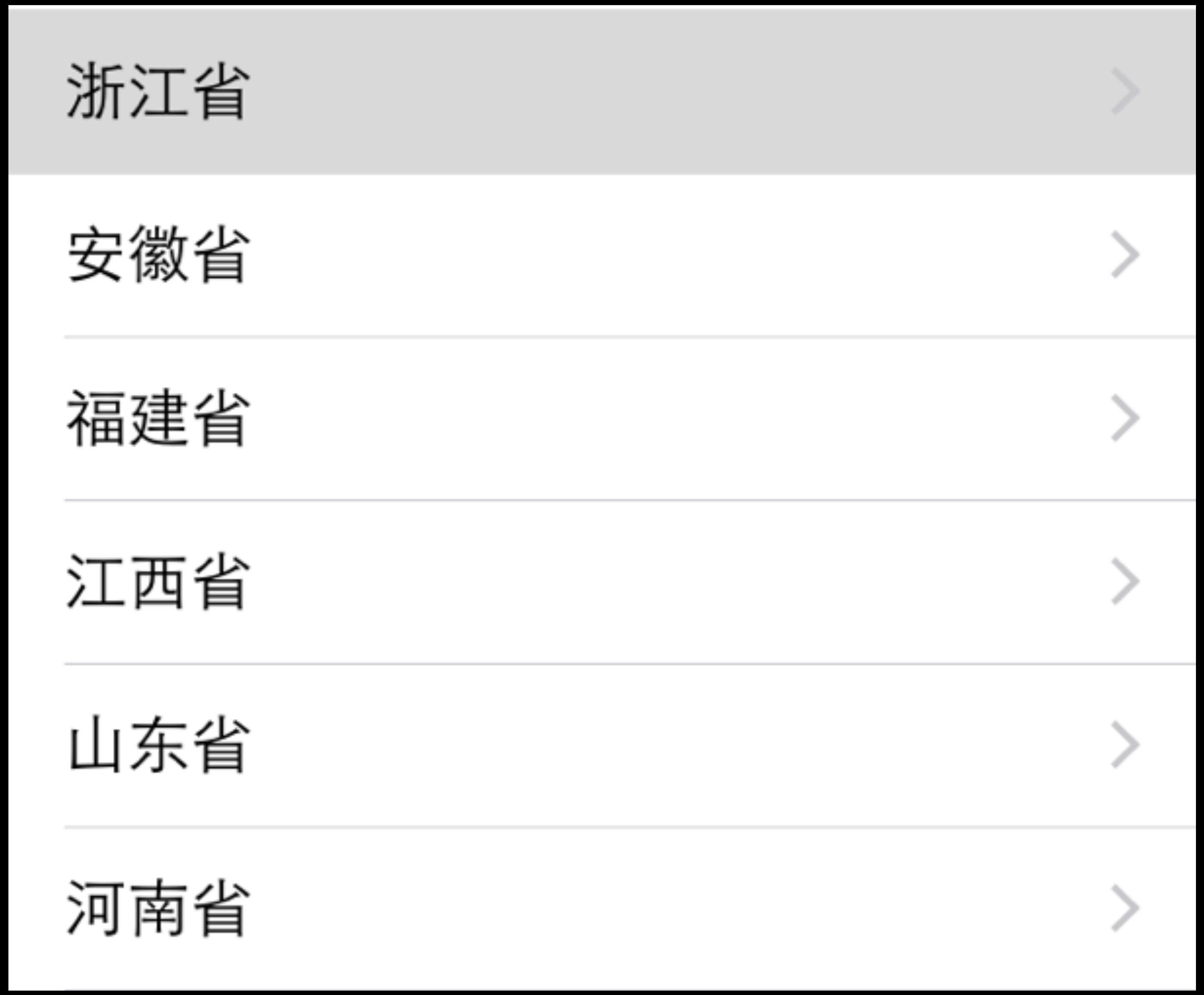
山东省

河南省

```
<UITableView>
    <UITableViewCell>浙江省</UITableViewCell>
    <UITableViewCell>安徽省</UITableViewCell>
    <UITableViewCell>福建省</UITableViewCell>
    <UITableViewCell>江西省</UITableViewCell>
    <UITableViewCell>山东省</UITableViewCell>
    <UITableViewCell>河南省</UITableViewCell>
<UITableView>
```

Simply, use <UITableViewCell/> instead of <div> ?

View-Core Solution



浙江省

安徽省

福建省

江西省

山东省

河南省

```
<UITableView data-path=“list”>
    <UITableViewCell>
        <UILabel data-path=“title”></UILabel>
    </UITableViewCell>
<UITableView>
```

✓

View-Core Solution

```
scope[“list”] = [
    { “title” : “浙江省” },
    { “title” : “安徽省” },
    { “title” : “福建省” },
    ...
]
```

➡

```
<UITableView name=“list”>
    <UITableViewCell>
        <UILabel name=“title”></UILabel>
    </UITableViewCell>
<UITableView>
```

✓

View-Core Solution

Github search ‘samurai-native’

gavinkwoe@github / qfish@github

Internal implementation

Internal implementation

• VSLRD Common Model	Define memory model
• Multiple Document Type	Define document model
• Reusable Render + Immutable DOM	Define render flow
• Store scope + Data serialization	Define data flow
• UIKit as a backend + Signal routing	Define event mechanism

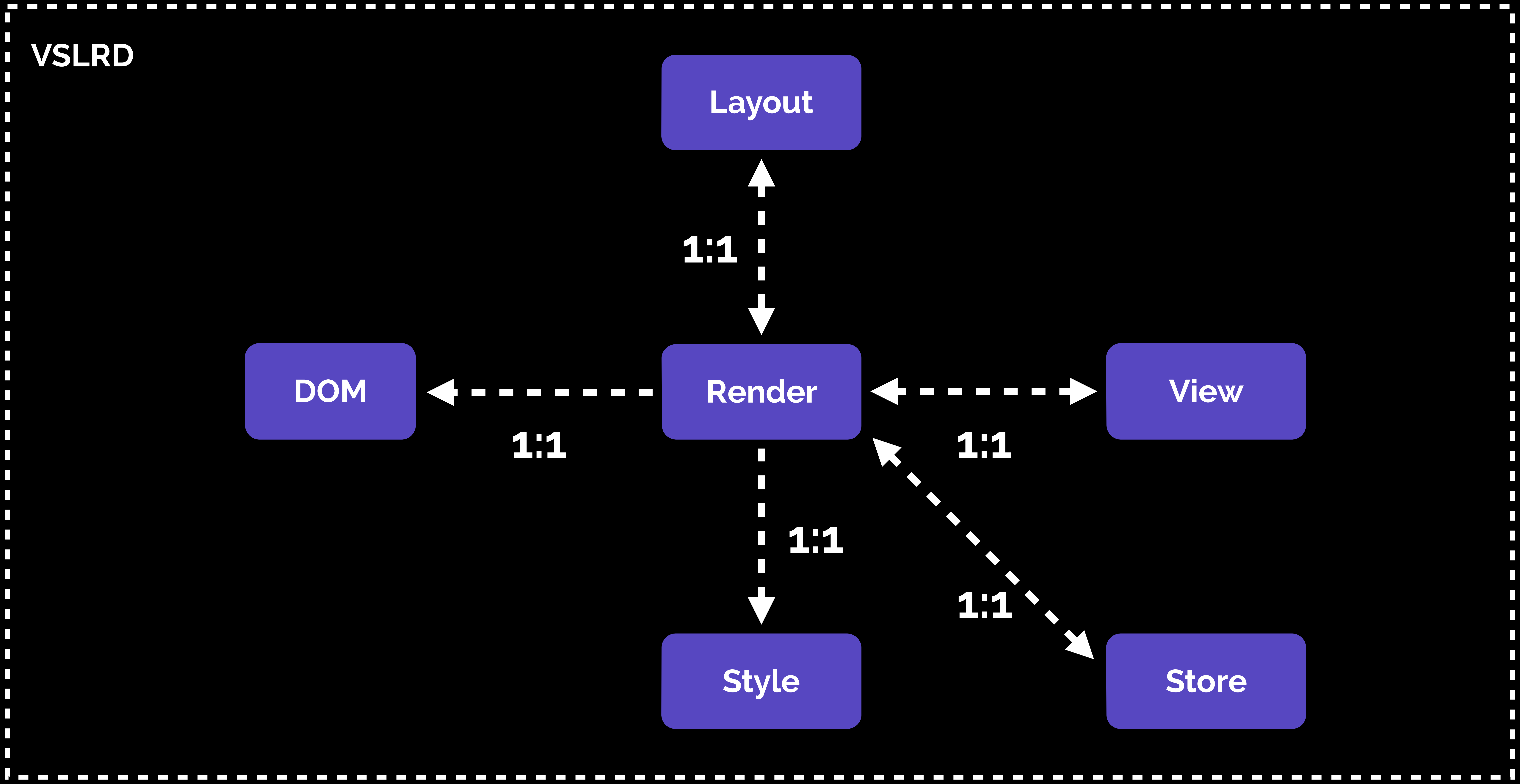
Internal implementation

- VSLRD Common Model
- Multiple Document Type
- Reusable Render + Immutable DOM
- Store scope + Data serialization
- UIKit as a backend + Signal routing

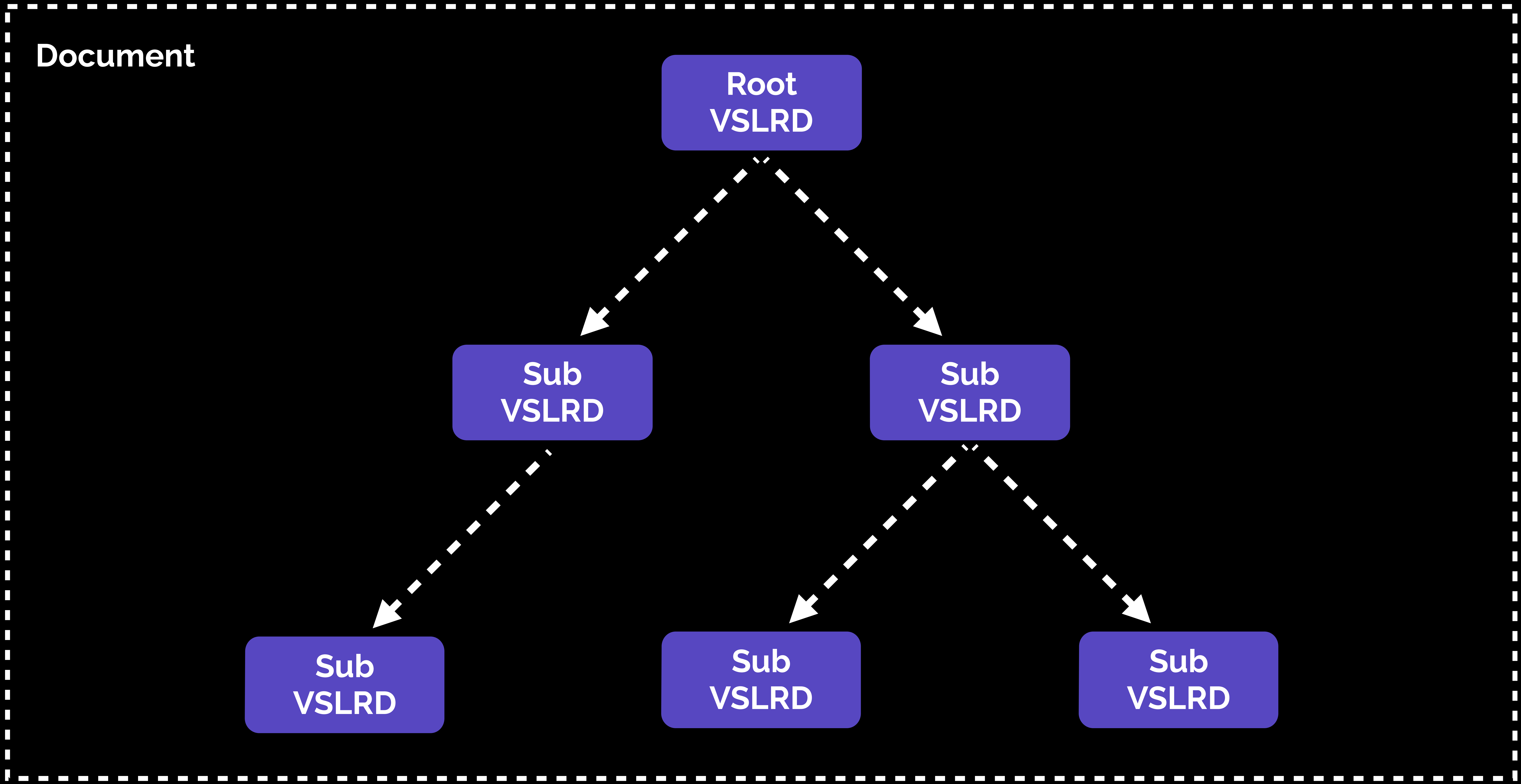
VSLRD Common Model

- View: Native view tree, UIKit
- Style: Style rules selected from stylesheet
- Layout: Abstract layout rules and algorithm
- Render: Render object + computed style
- StoreScope: Data binding with node context
- DOM/Document

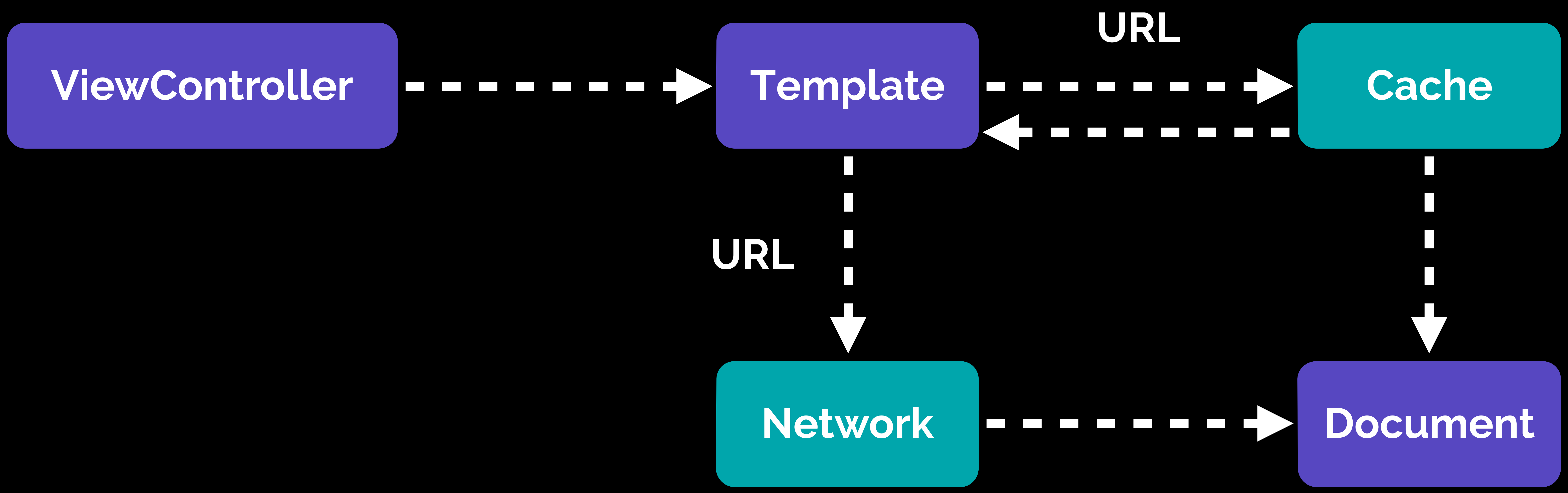
VSLRD Common Model



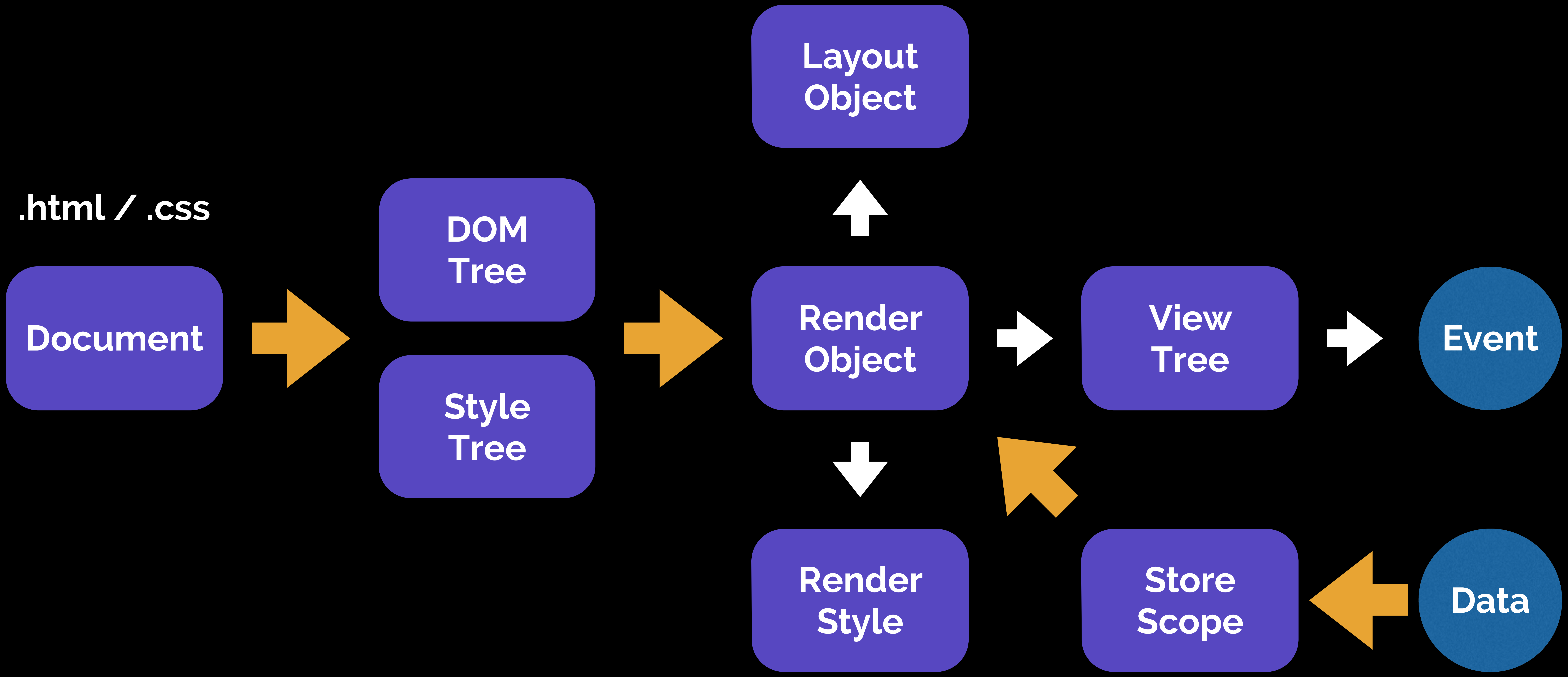
VSLRD Common Model



VSLRD Common Model



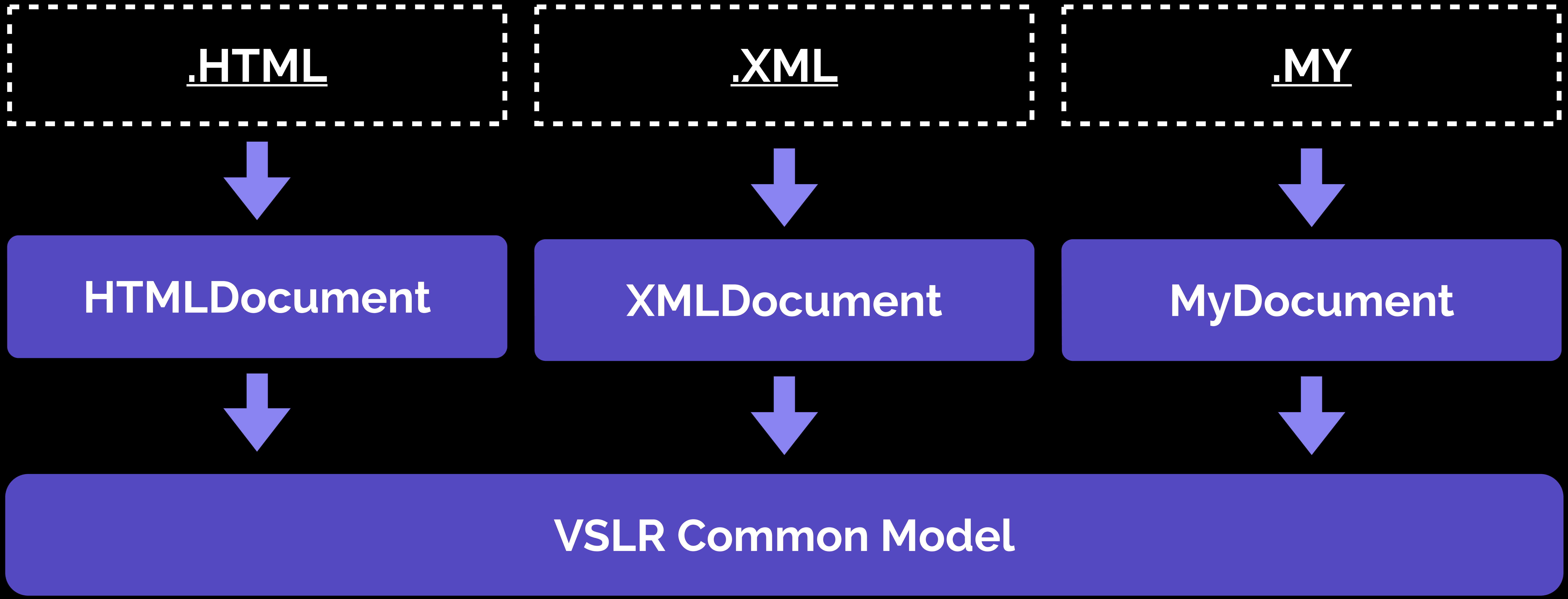
VSLRD Common Model



Internal implementation

- VSLRD Common Model
- Multiple Document Type
- Reusable Render + Immutable DOM
- Store scope + Data serialization
- UIKit as a backend + Signal routing

Multiple Document Type


.HTML

.XML

.MY

HTMLDocument

XMLDocument

MyDocument

VSLR Common Model

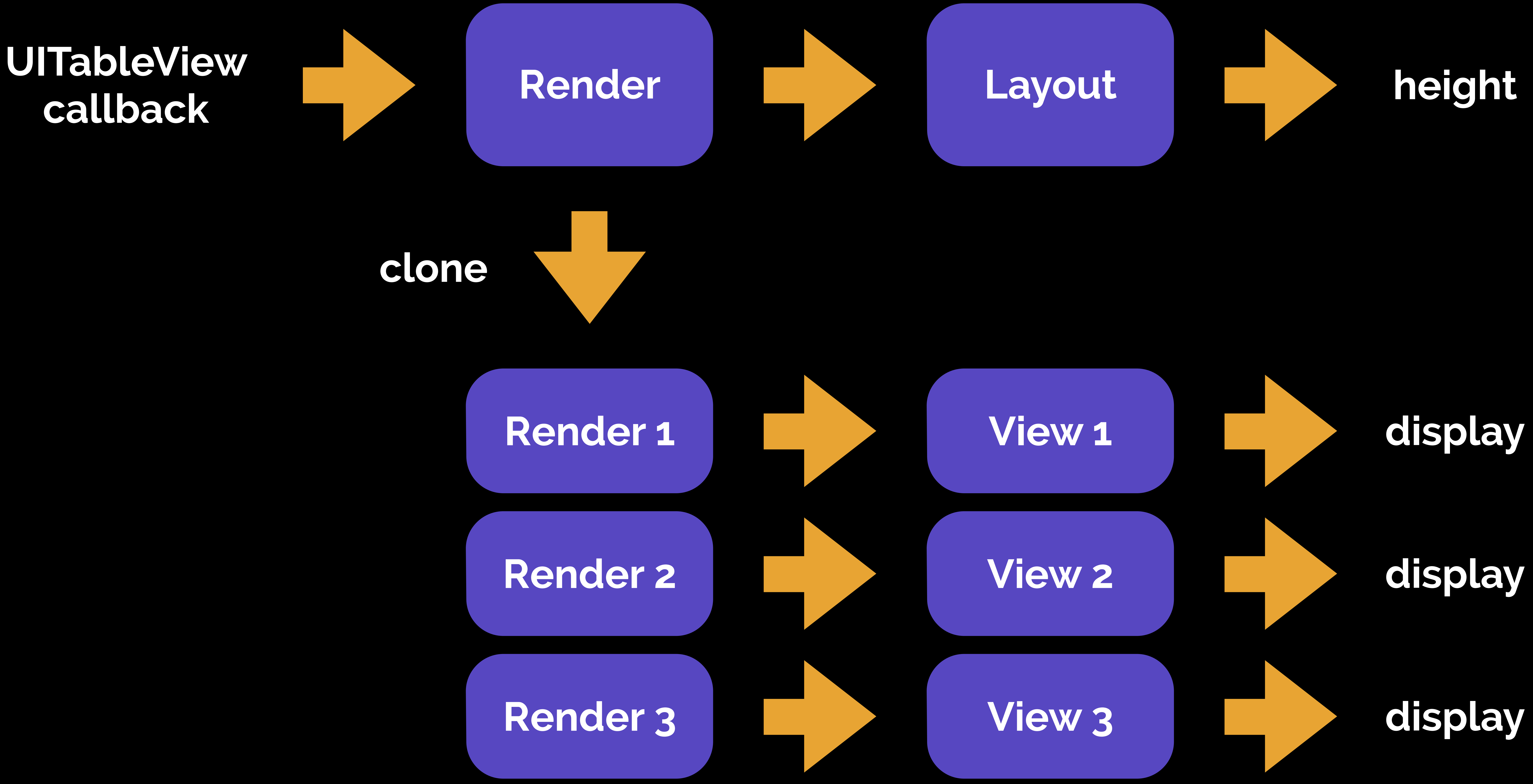
Multiple Document Type

- HTML Document
  - HTML DOM
  - HTML Render
  - HTML RenderStyle
  - HTML Layout
  - HTML Store
  - HTML Elements & Native Components

Internal implementation

- VSLRD Common Model
- Multiple Document Type
- Reusable Render + Immutable DOM
- Store scope + Data serialization
- UIKit as a backend + Signal routing

Reusable Render


UITableView callback

Render

Layout

height

clone

Render 1

View 1

display

Render 2

View 2

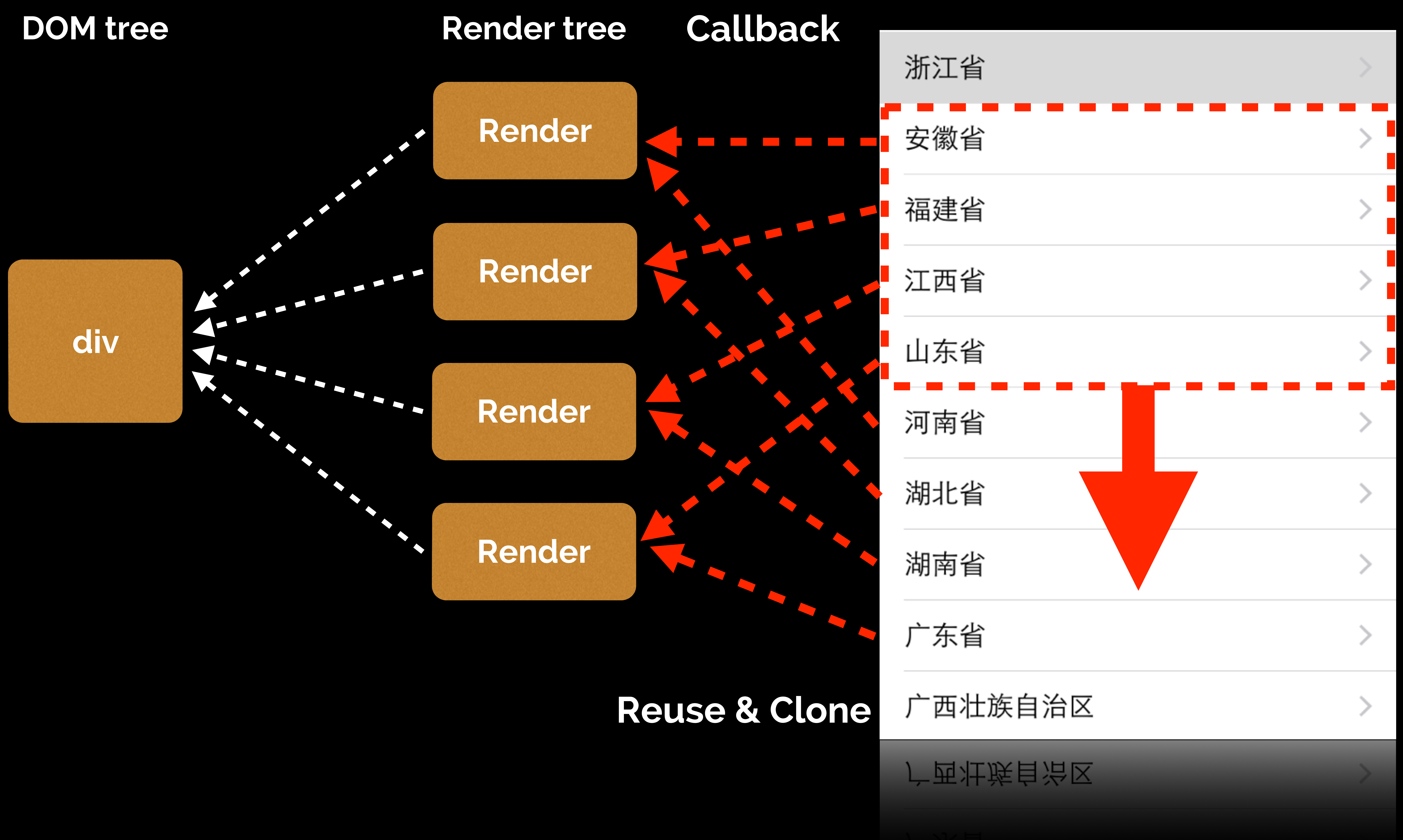
display

Render 3

View 3

display

Reusable Render



浙江省

安徽省

福建省

江西省

山东省

河南省

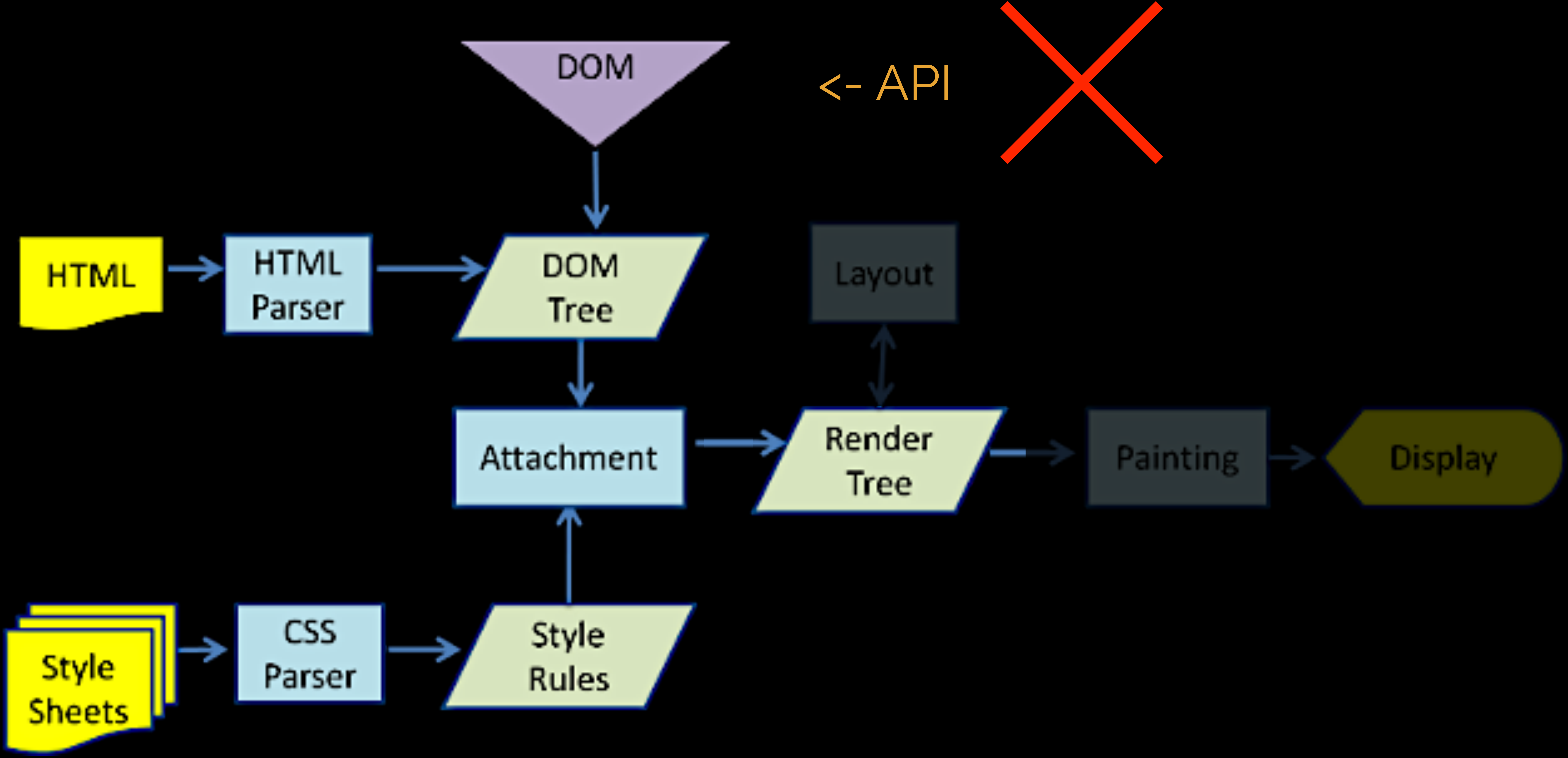
湖北省

湖南省

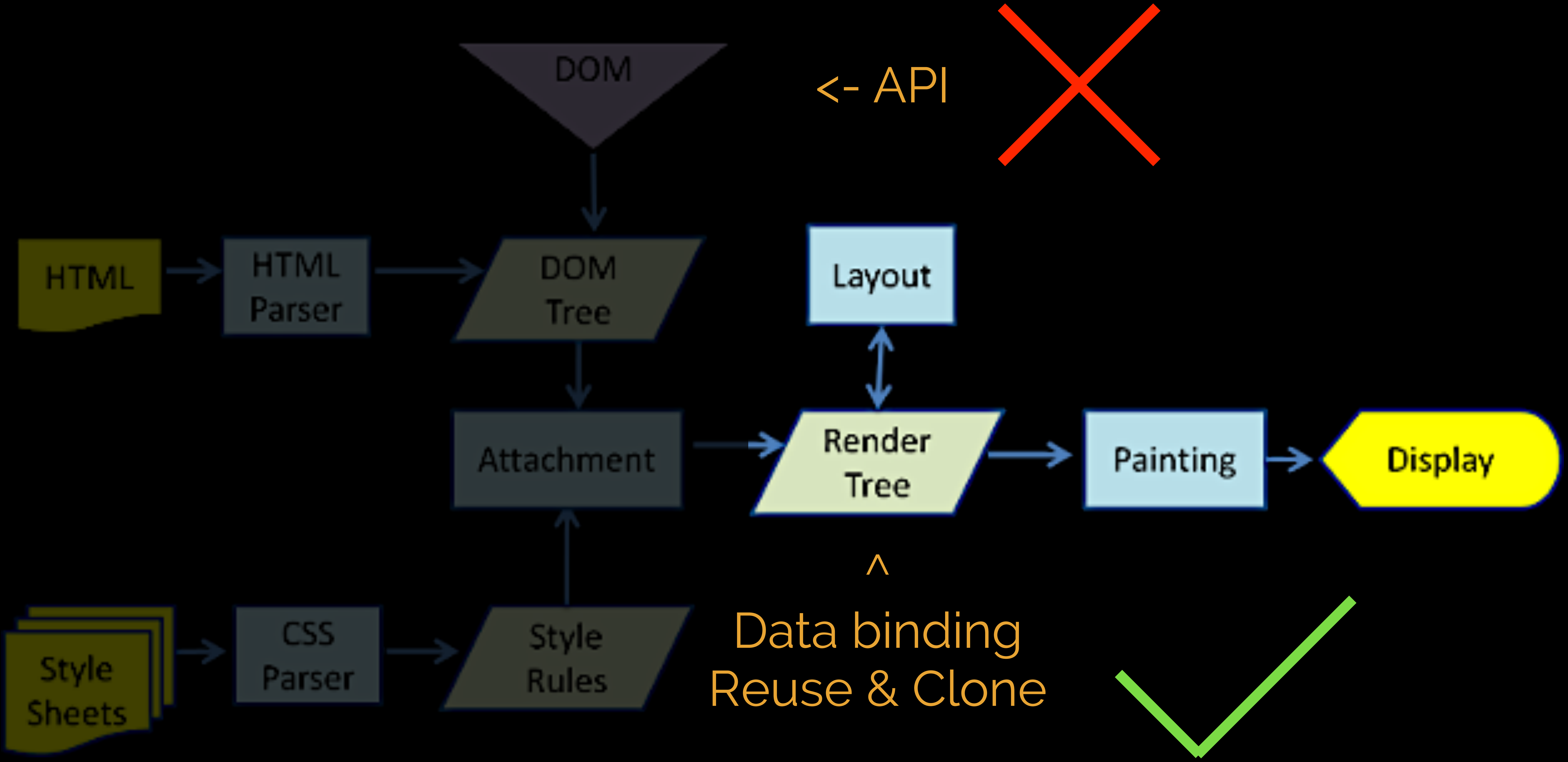
广东省

广西壮族自治区

Immutable DOM



Immutable DOM



DOM

<- API

HTML

HTML Parser

DOM Tree

Layout

Attachment

Render Tree

Painting

Display

Style Sheets

CSS Parser

Style Rules

^

Data binding
Reuse & Clone

Internal implementation

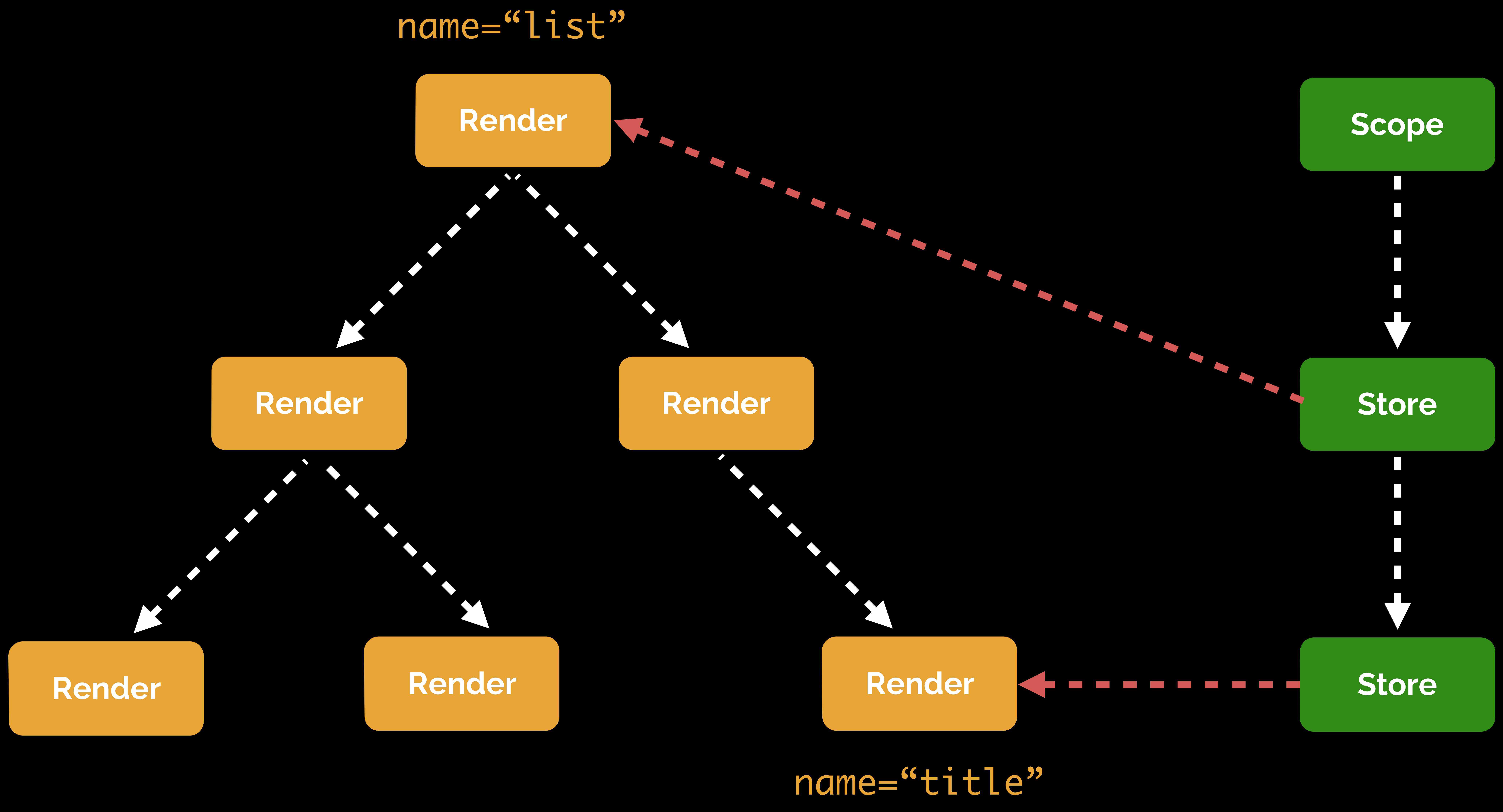
- VSLRD Common Model
- Multiple Document Type
- Reusable Render + Immutable DOM
- Store scope + Data serialization
- UIKit as a backend + Signal routing

Store scope

```
scope[“list”] = [
    { “title” : “浙江省” },
    { “title” : “安徽省” },
    { “title” : “福建省” },
    ...
]
```

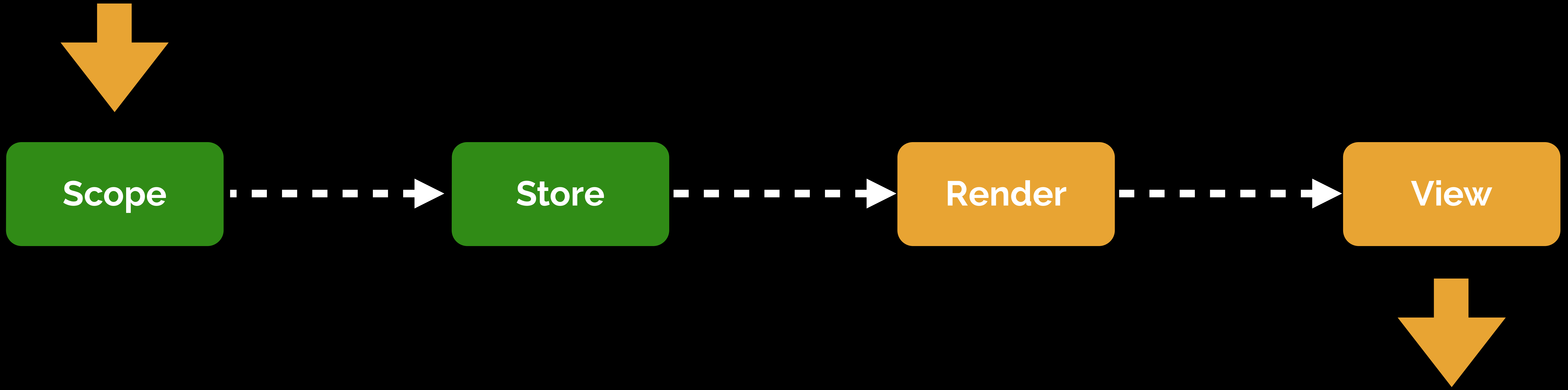
```
<UITableView name=“list”>
    <UITableViewCell>
        <UILabel name=“title”></UILabel>
    </UITableViewCell>
<UITableView>
```

Store scope


name="list"
Render
Render
Render
Render
Render
Render
Scope
Store
Store
name="title"

Data serialization

```
[
    { “title” : “浙江省” },
    { “title” : “安徽省” }
]
```



```
- (id)serialize;
- (void)unserialize:(id)obj;
```

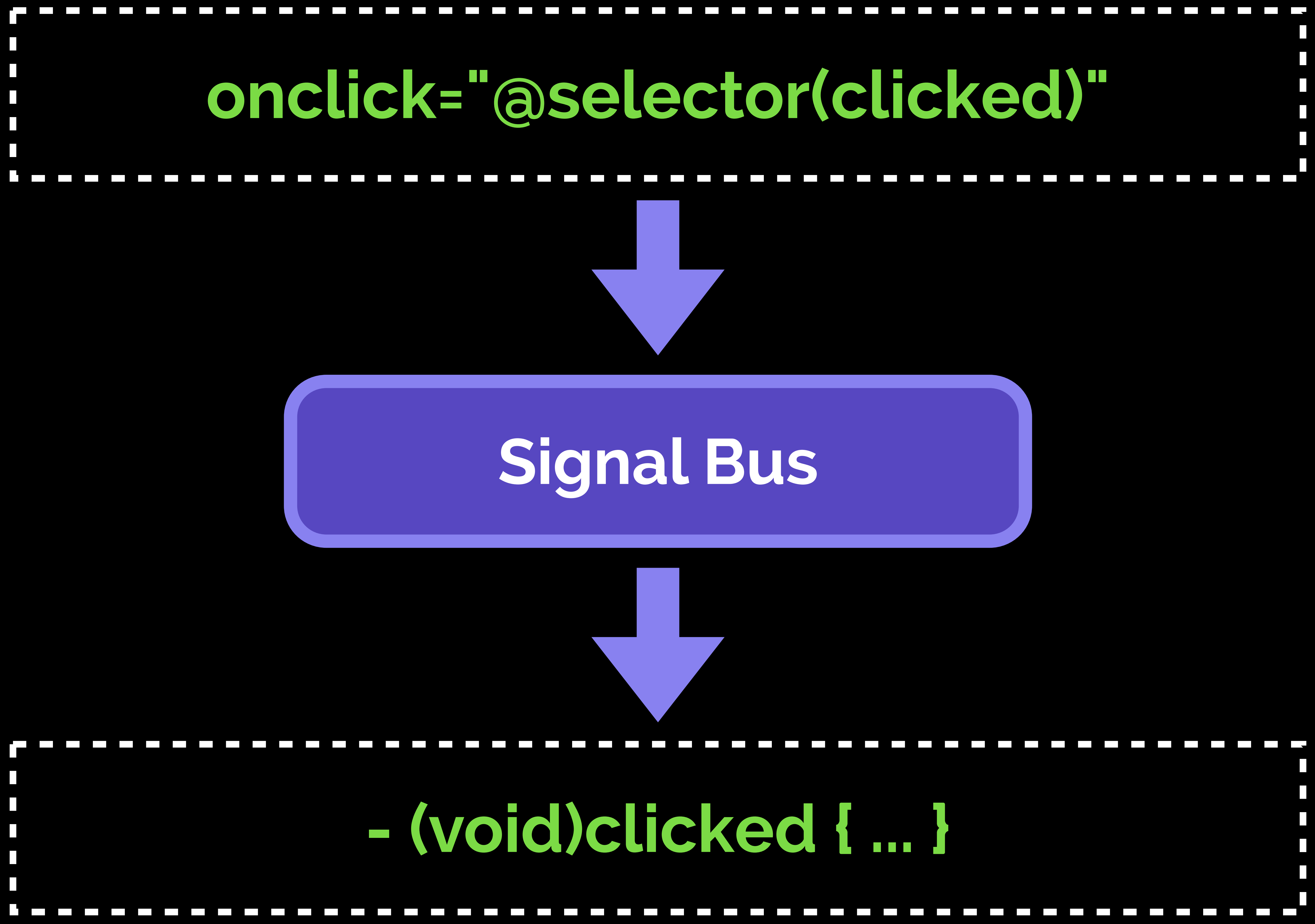
Internal implementation

- VSLRD Common Model
- Multiple Document Type
- Reusable Render + Immutable DOM
- Store scope + Data serialization
- UIKit as a backend + Signal routing

UIKit as a backend

- Native gesture & responder chain
- Native user experience
- Native UI control library
- System decide when to painting

Signal routing

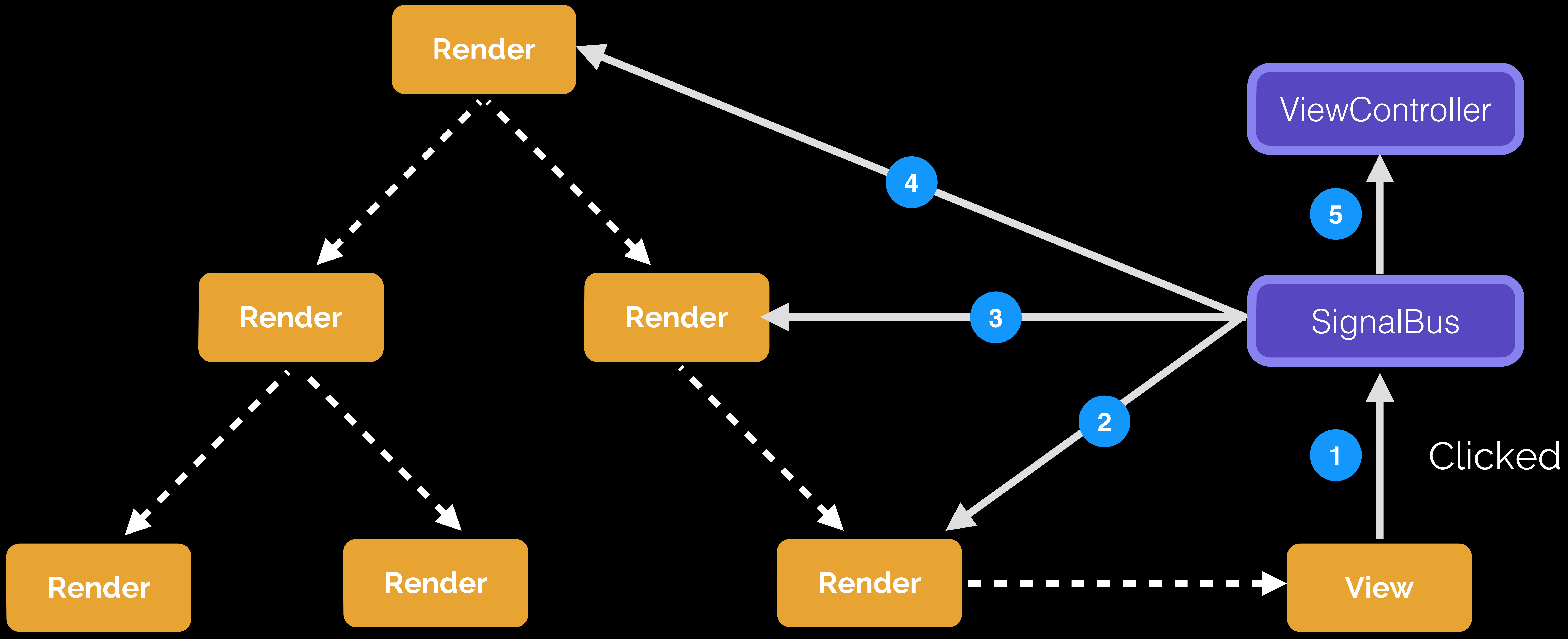


onclick="@selector(clicked)"

Signal Bus

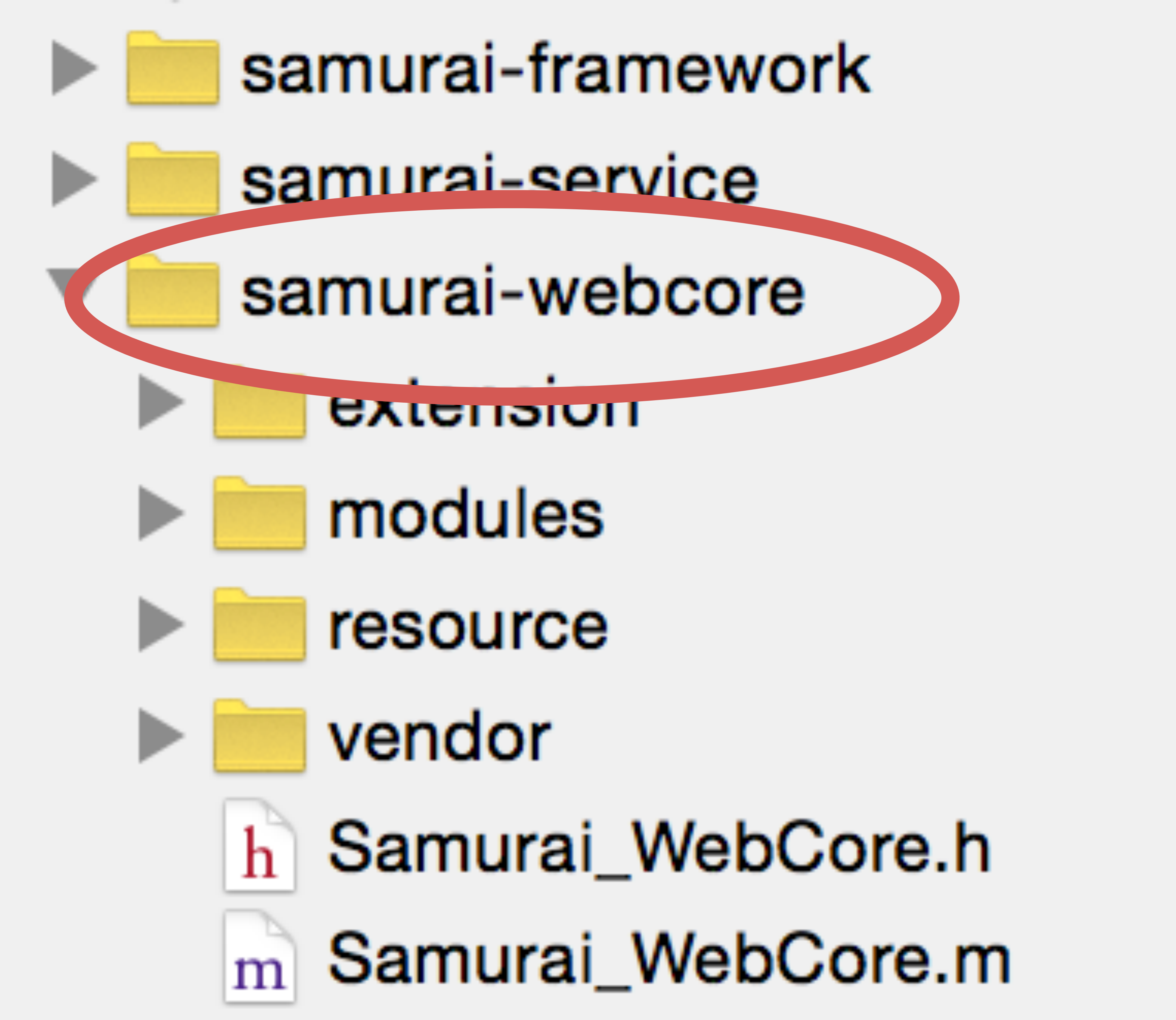
- (void)clicked { ... }

Bubbling mechanism

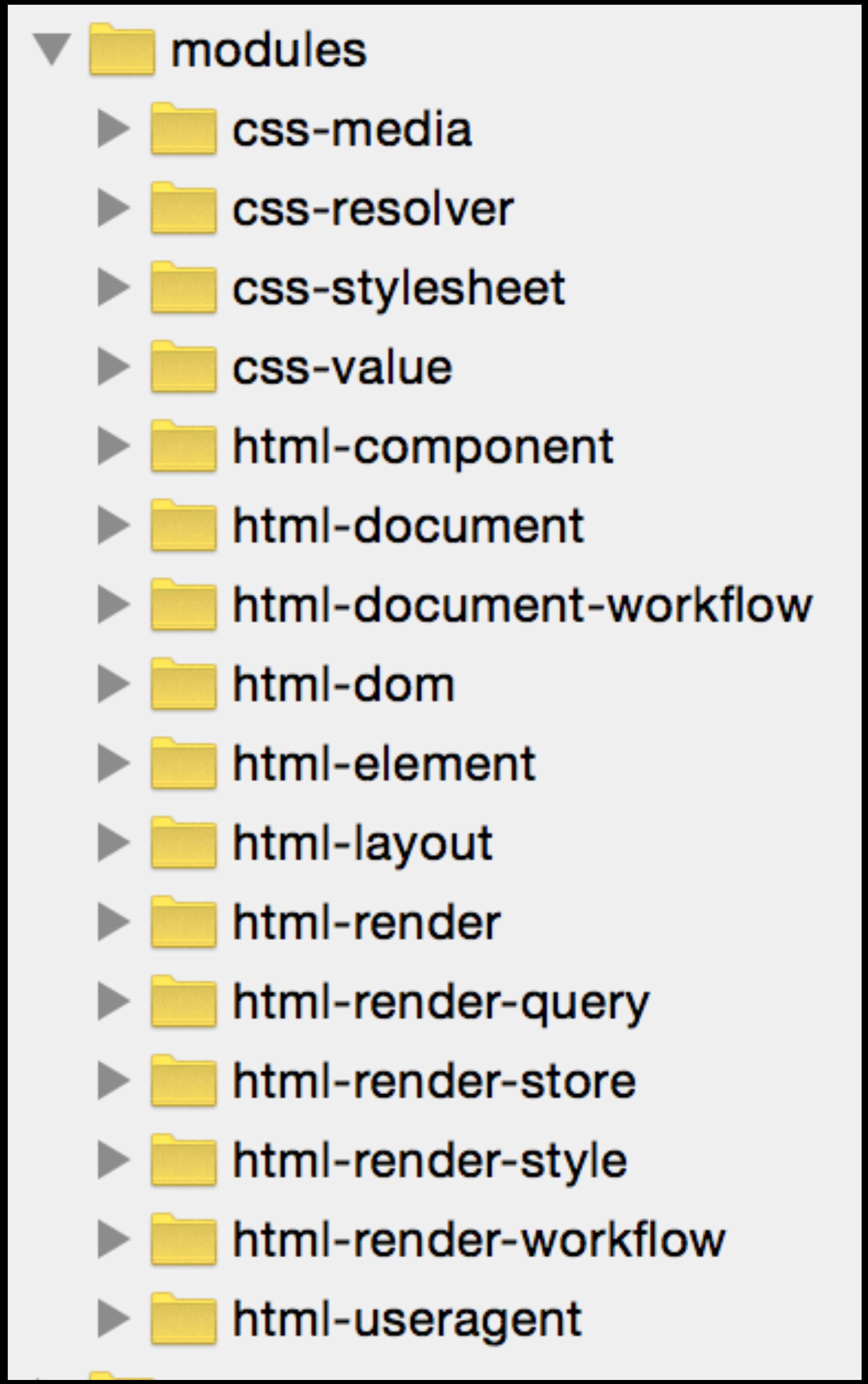


Do it yourself, My-Core

Project directory



```
▶ samurai-framework
▶ samurai-service
▼ samurai-webcore
    ▶ extension
    ▶ modules
    ▶ resource
    ▶ vendor
      Samurai_WebCore.h
      Samurai_WebCore.m
```

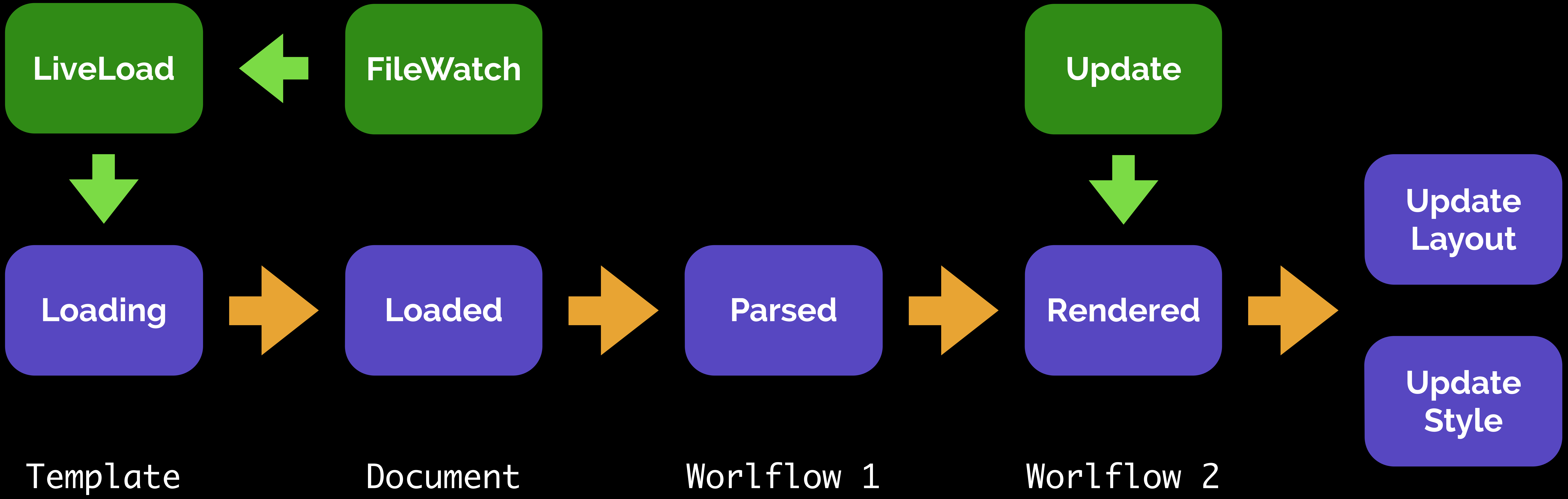


```
▼ modules
    ▶ css-media
    ▶ css-resolver
    ▶ css-stylesheet
    ▶ css-value
    ▶ html-component
    ▶ html-document
    ▶ html-document-workflow
    ▶ html-dom
    ▶ html-element
    ▶ html-layout
    ▶ html-render
    ▶ html-render-query
    ▶ html-render-store
    ▶ html-render-style
    ▶ html-render-workflow
    ▶ html-useragent
```

MUST HAVE Class

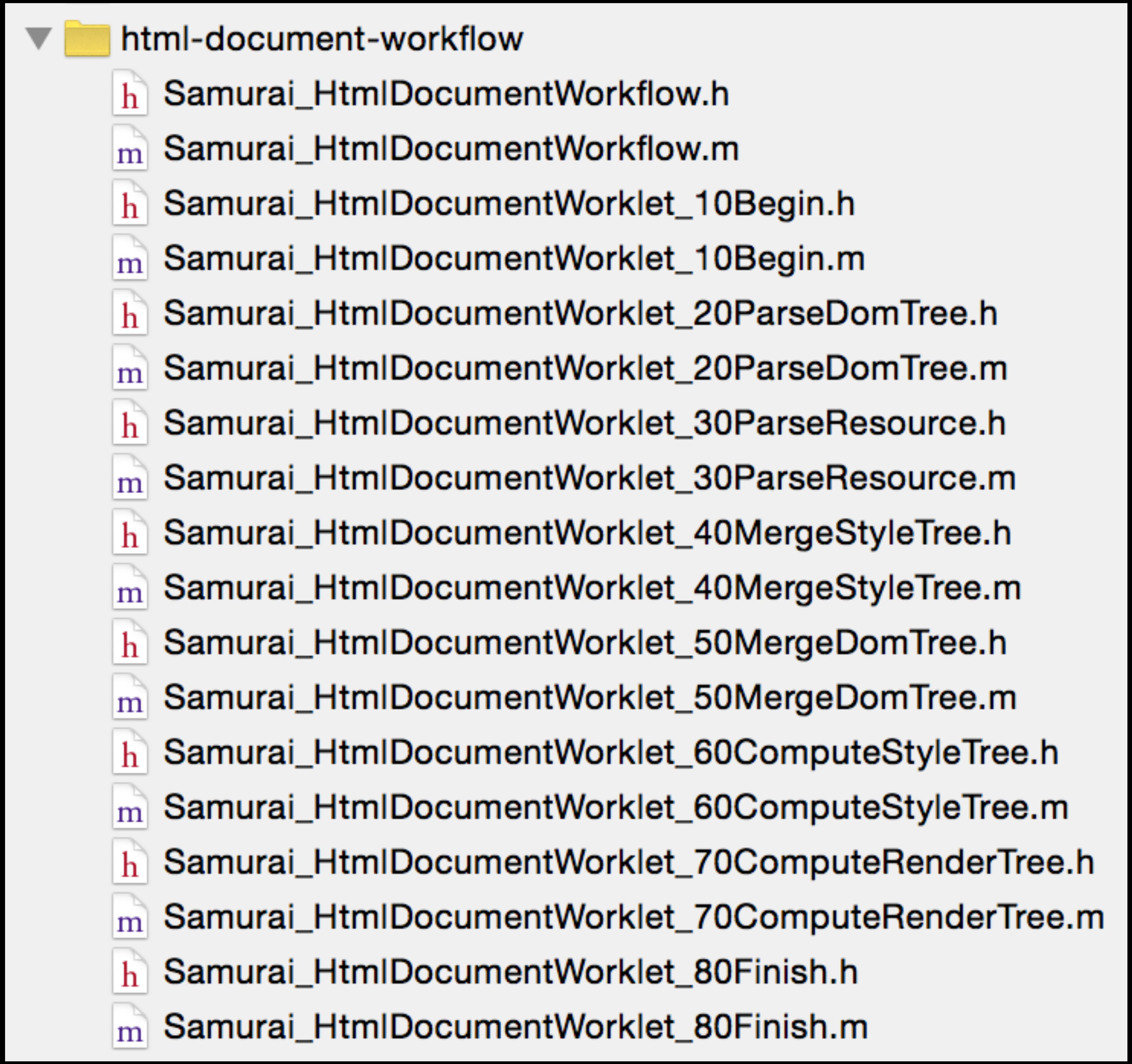
- MyDOM : SamuraiDOM
- MyRender : SamuraiRenderObject
- MyRenderStyle : SamuraiRenderStyle
- MyLayout : SamuraiLayoutObject
- MyScope : SamuraiRenderScope

MUST HAVE Workflow



MUST HAVE Workflow 1/2

- Document workflow
  - 10 Parse DOM Tree
  - 20 Parse Resource
  - 30 Merge Style Tree
  - 40 Merge DOM Tree
  - 50 Compute Style Tree
  - 60 Compute Render tree

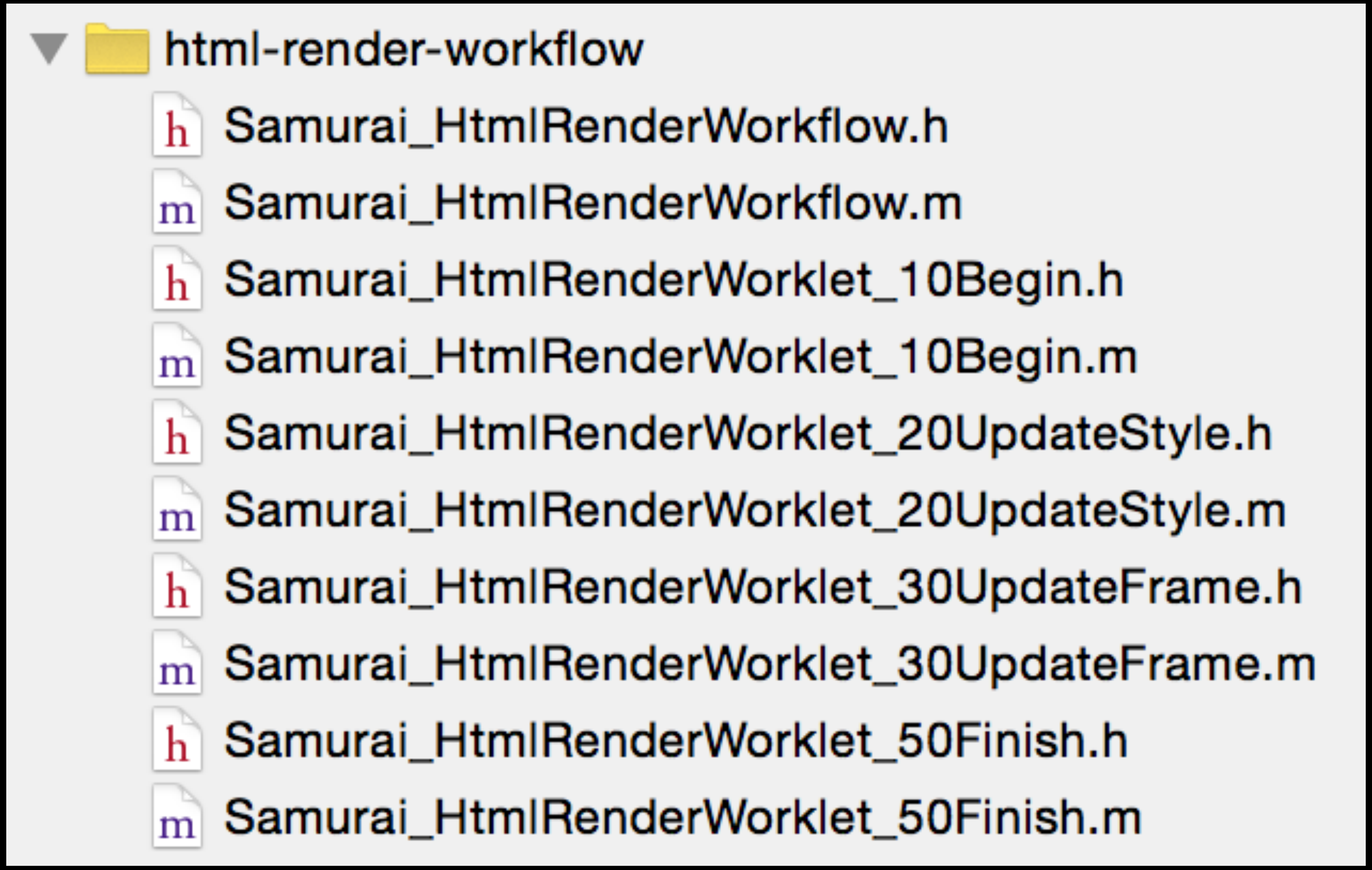


html-document-workflow

- Samurai_HtmlDocumentWorkflow.h
- Samurai_HtmlDocumentWorkflow.m
- Samurai_HtmlDocumentWorklet_10Begin.h
- Samurai_HtmlDocumentWorklet_10Begin.m
- Samurai_HtmlDocumentWorklet_20ParseDomTree.h
- Samurai_HtmlDocumentWorklet_20ParseDomTree.m
- Samurai_HtmlDocumentWorklet_30ParseResource.h
- Samurai_HtmlDocumentWorklet_30ParseResource.m
- Samurai_HtmlDocumentWorklet_40MergeStyleTree.h
- Samurai_HtmlDocumentWorklet_40MergeStyleTree.m
- Samurai_HtmlDocumentWorklet_50MergeDomTree.h
- Samurai_HtmlDocumentWorklet_50MergeDomTree.m
- Samurai_HtmlDocumentWorklet_60ComputeStyleTree.h
- Samurai_HtmlDocumentWorklet_60ComputeStyleTree.m
- Samurai_HtmlDocumentWorklet_70ComputeRenderTree.h
- Samurai_HtmlDocumentWorklet_70ComputeRenderTree.m
- Samurai_HtmlDocumentWorklet_80Finish.h
- Samurai_HtmlDocumentWorklet_80Finish.m

MUST HAVE Workflow 2/2

- Render workflow
  - 10 Update style
  - 20 Update frame



```
html-render-workflow
    Samurai_HtmlRenderWorkflow.h
    Samurai_HtmlRenderWorkflow.m
    Samurai_HtmlRenderWorklet_10Begin.h
    Samurai_HtmlRenderWorklet_10Begin.m
    Samurai_HtmlRenderWorklet_20UpdateStyle.h
    Samurai_HtmlRenderWorklet_20UpdateStyle.m
    Samurai_HtmlRenderWorklet_30UpdateFrame.h
    Samurai_HtmlRenderWorklet_30UpdateFrame.m
    Samurai_HtmlRenderWorklet_50Finish.h
    Samurai_HtmlRenderWorklet_50Finish.m
```

Define document

- MyDocument : SamuraiDocument
  - + (NSArray *)supportedExtensions { return @[**@“html”**]; }
  - + (NSArray *)supportedTypes { return @[**@“text/html”**]; }
  - - (BOOL)parse { … } **call Worlflow 1 to re-parse**
  - - (BOOL)reflow { … } **call Worlflow 2 to re-style/re-layout**

Test

- @implementation MyViewController
    - [self **loadTemplate:@“xxx.my”];**

More implementation detail

Tag support

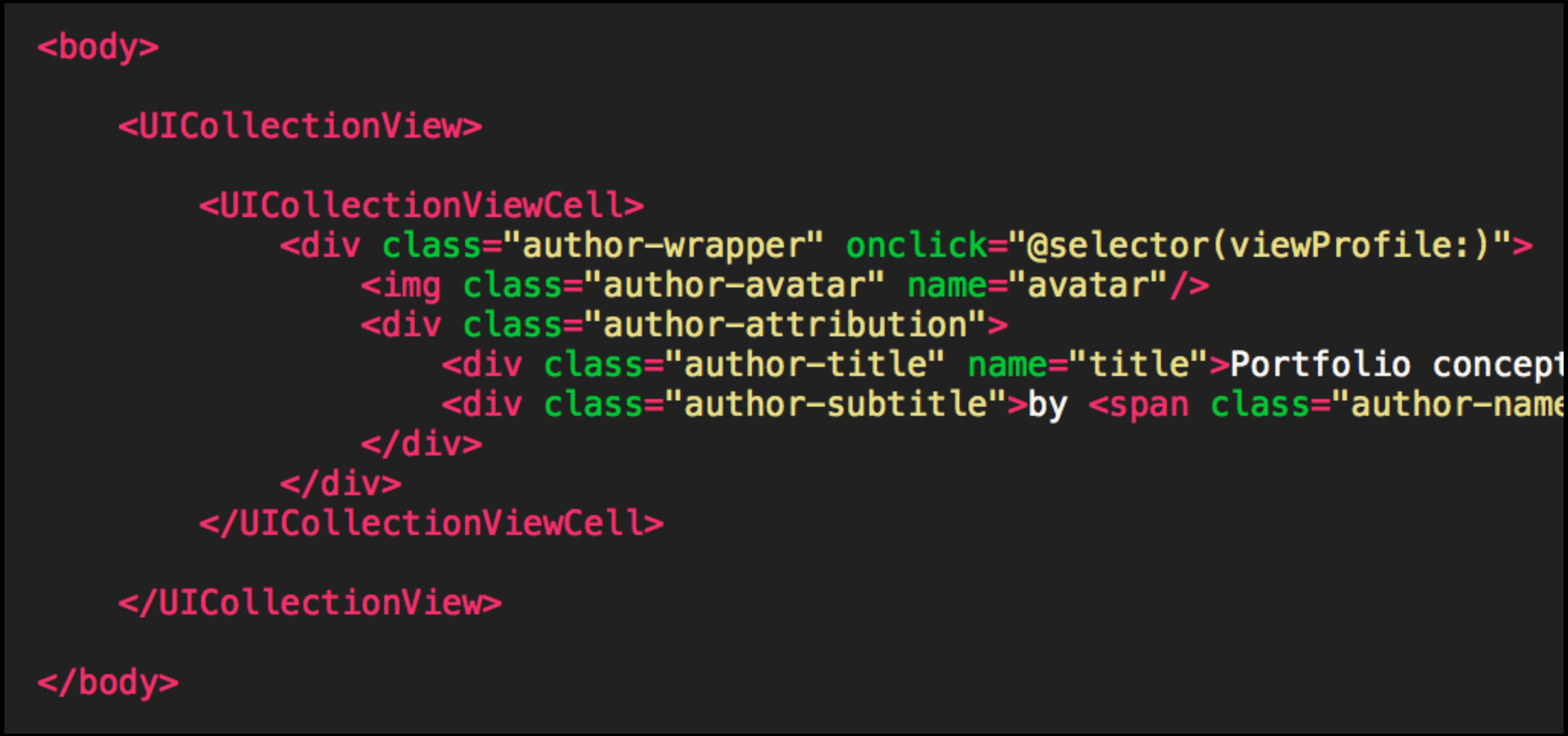
20+ UIKit Components

+

70+ HTML Elements

Support UITableView and UICollectionView

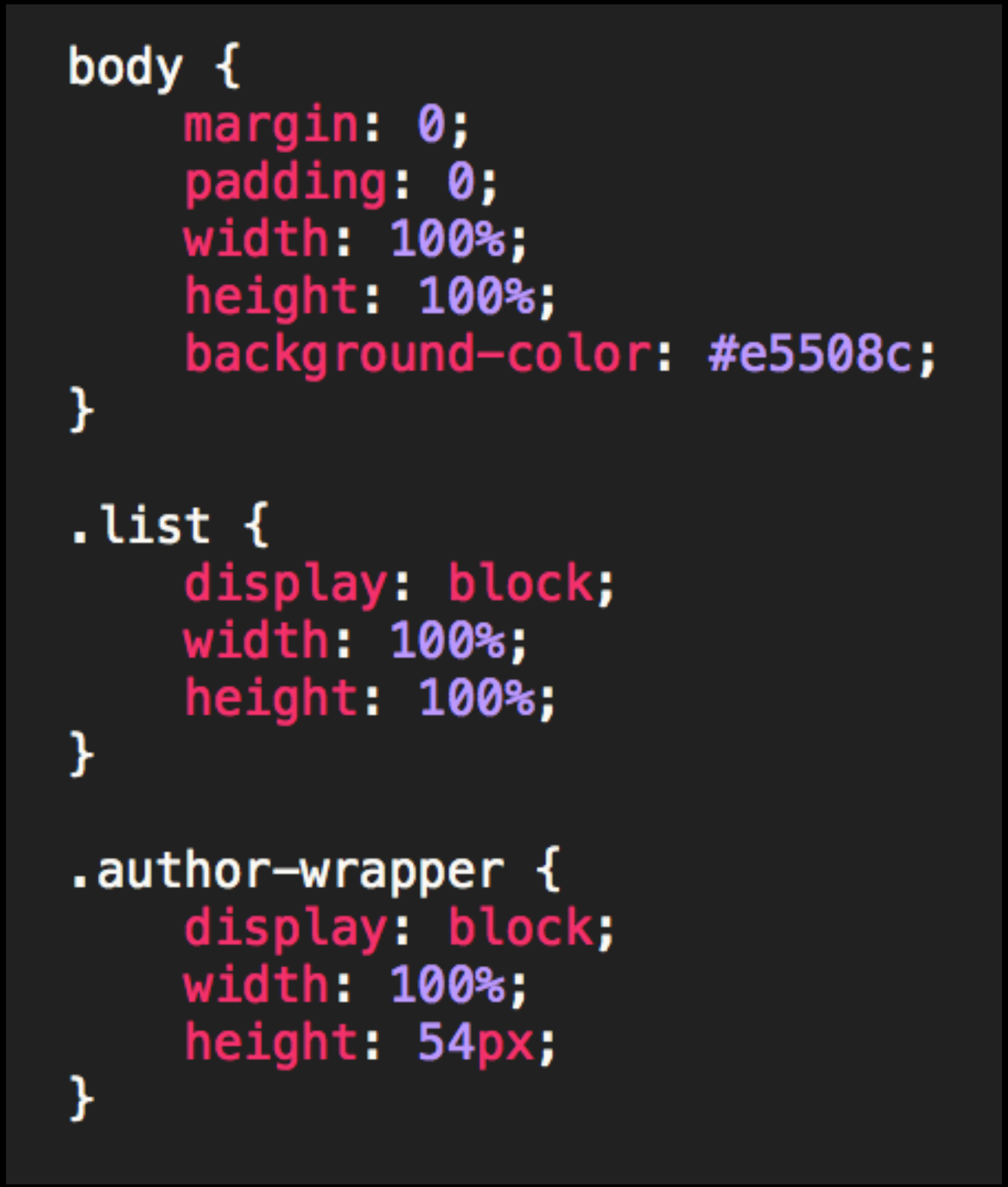
UIML standard support



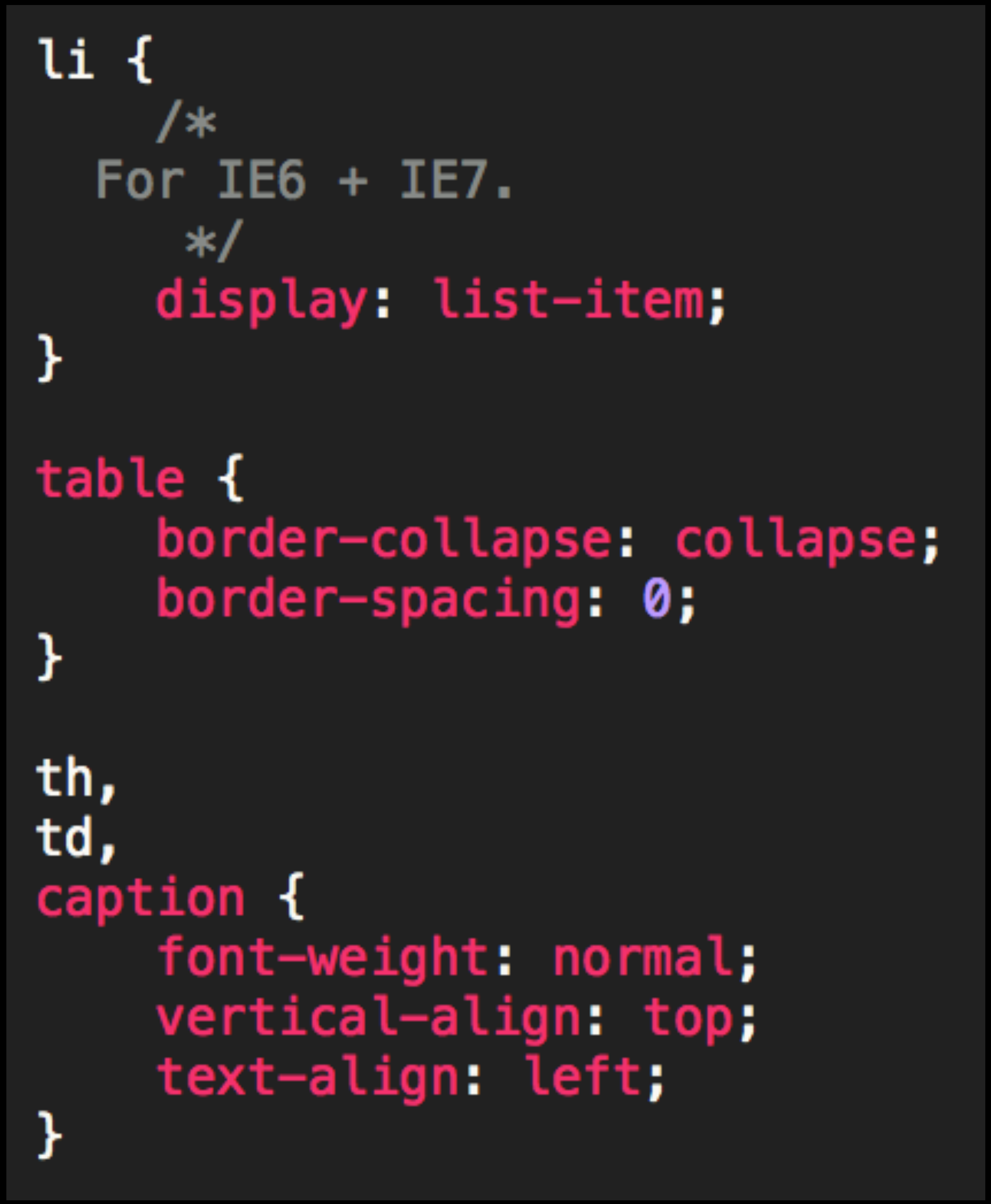
```
<body>

    <UICollectionView>

        <UICollectionViewCell>
            <div class="author-wrapper" onclick="@selector(viewProfile:)">
                <img class="author-avatar" name="avatar"/>
                <div class="author-attribution">
                    <div class="author-title" name="title">Portfolio concept
                    <div class="author-subtitle">by <span class="author-name
                </div>
            </div>
        </UICollectionViewCell>

    </UICollectionView>

</body>
```

Standard HTML grammer

CSS standard support



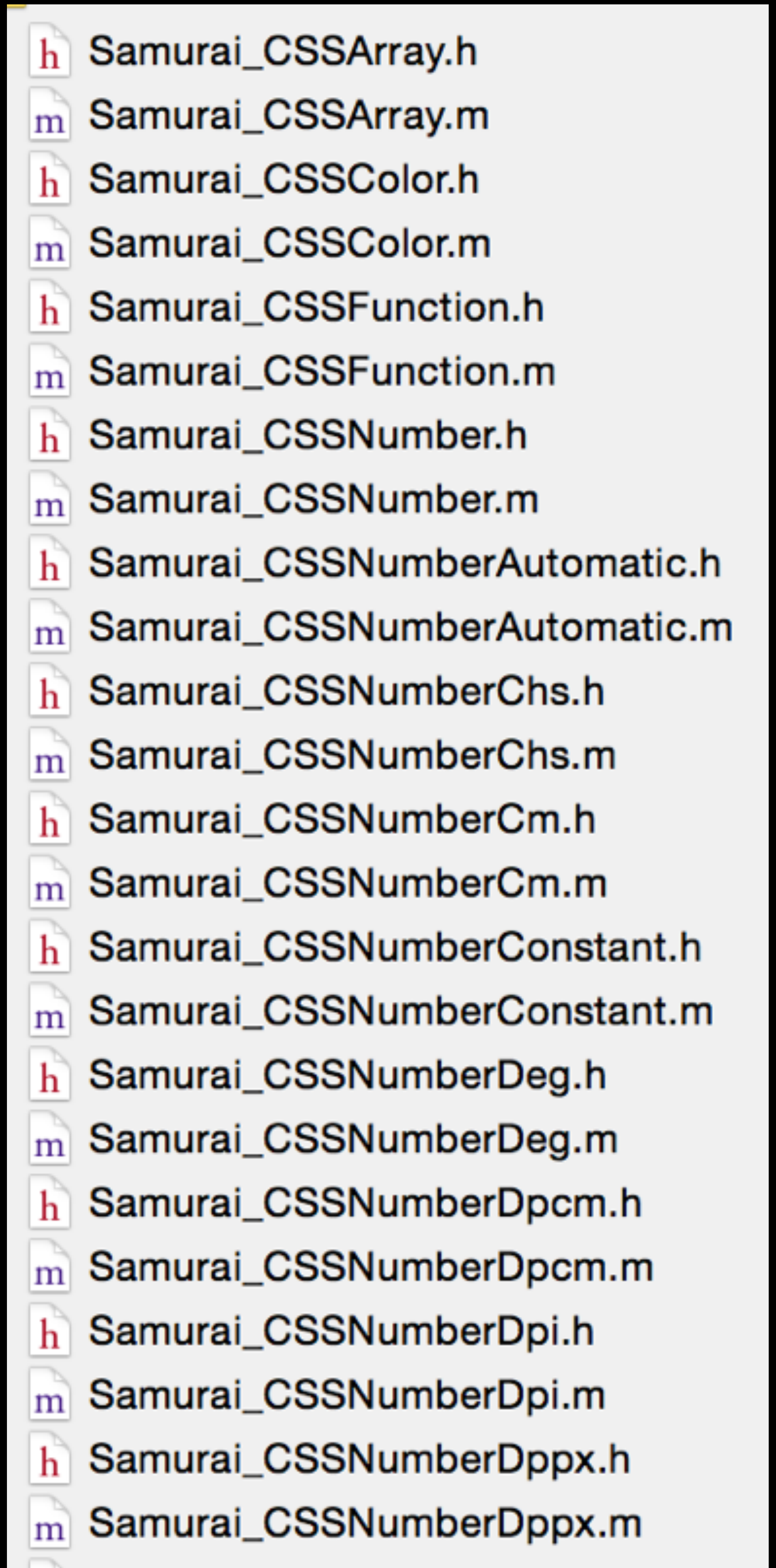
```
body {
    margin: 0;
    padding: 0;
    width: 100%;
    height: 100%;
    background-color: #e5508c;
}

.list {
    display: block;
    width: 100%;
    height: 100%;
}

.author-wrapper {
    display: block;
    width: 100%;
    height: 54px;
}
```



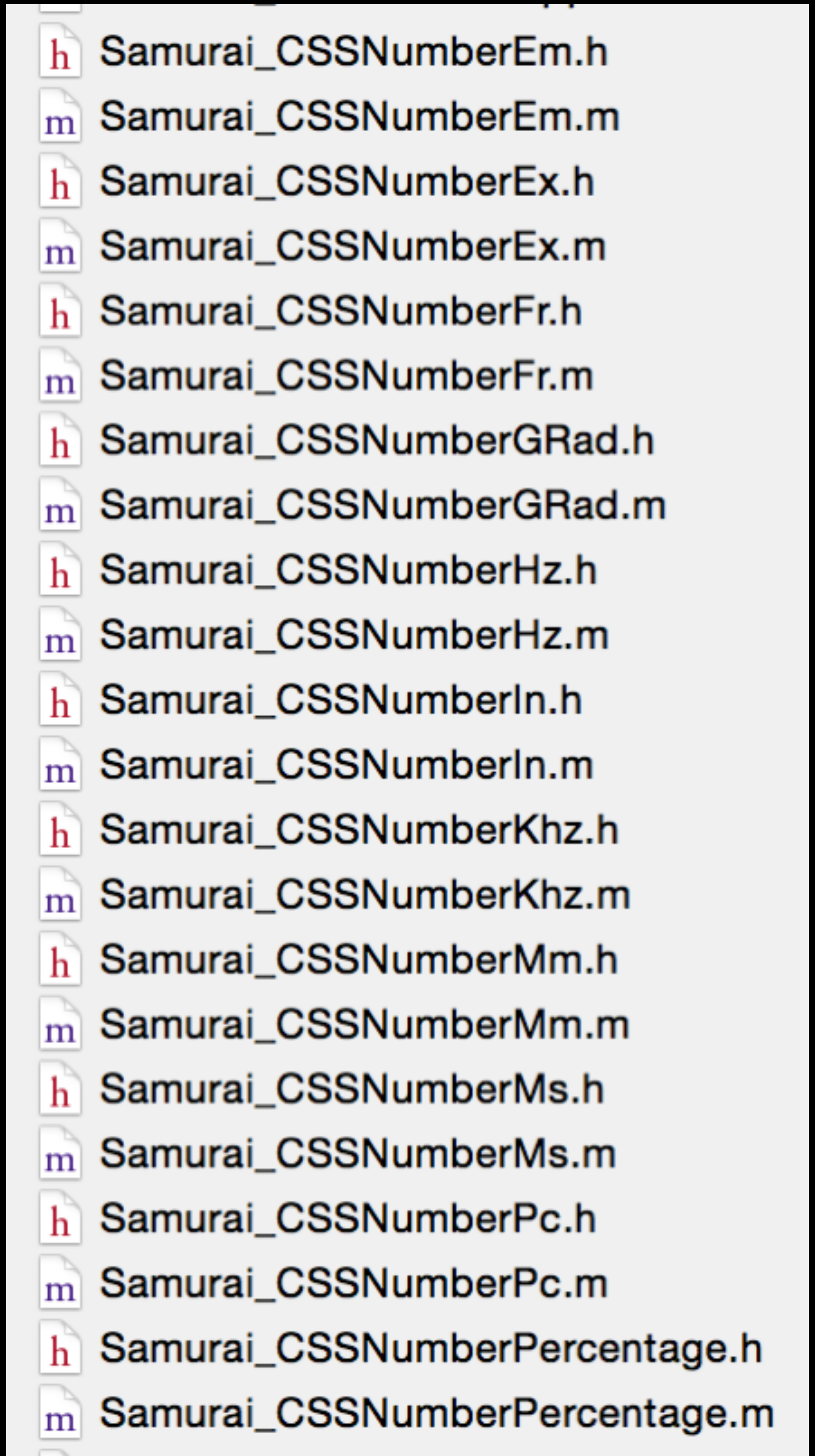
```
li {
    /*
  For IE6 + IE7.
     */
    display: list-item;
}

table {
    border-collapse: collapse;
    border-spacing: 0;
}

th,
td,
caption {
    font-weight: normal;
    vertical-align: top;
    text-align: left;
}
```

Standard fluid layout, standard box model

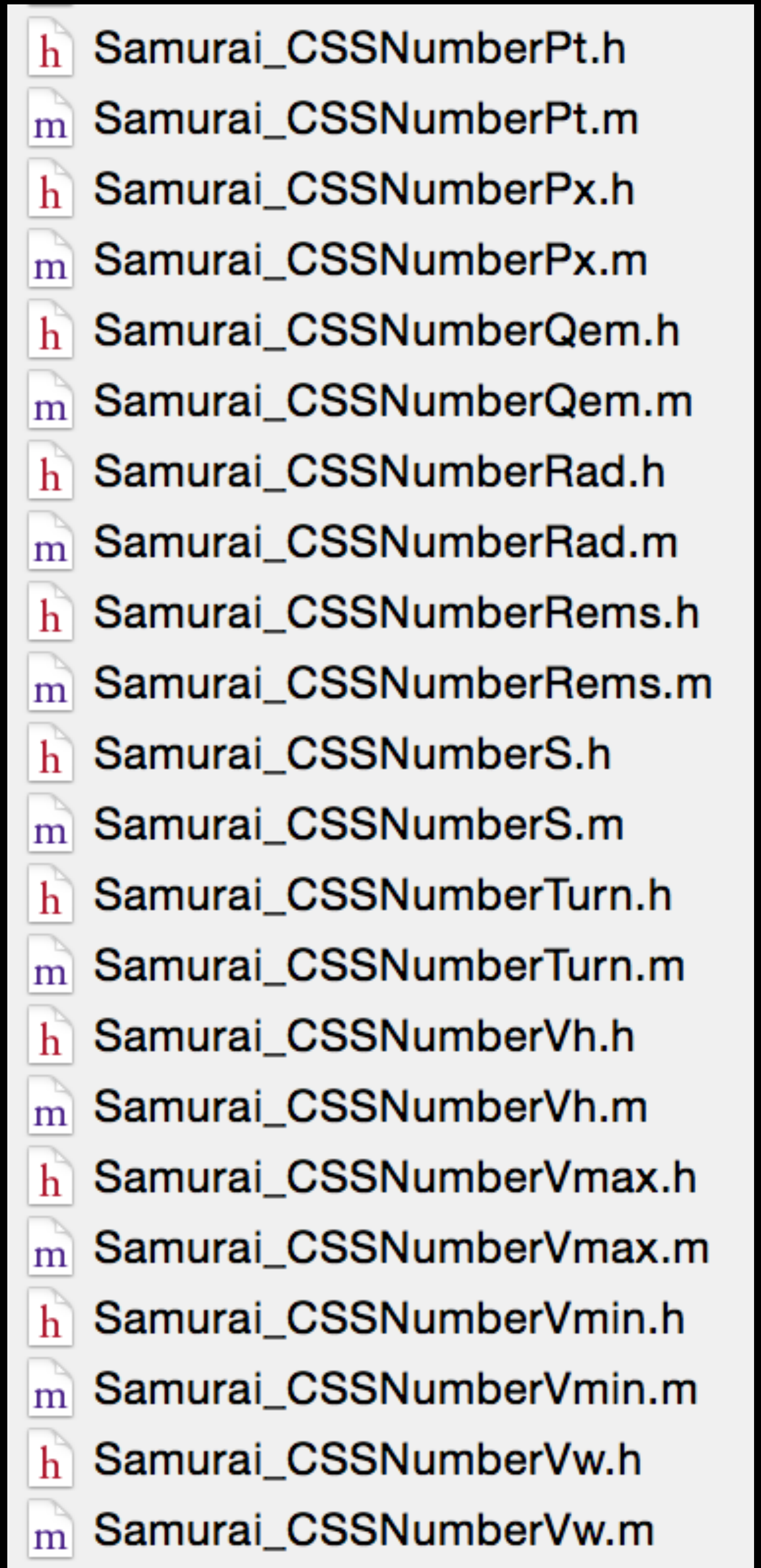
CSS unit support



Samurai_CSSArray.h
Samurai_CSSArray.m
Samurai_CSSColor.h
Samurai_CSSColor.m
Samurai_CSSFunction.h
Samurai_CSSFunction.m
Samurai_CSSNumber.h
Samurai_CSSNumber.m
Samurai_CSSNumberAutomatic.h
Samurai_CSSNumberAutomatic.m
Samurai_CSSNumberChs.h
Samurai_CSSNumberChs.m
Samurai_CSSNumberCm.h
Samurai_CSSNumberCm.m
Samurai_CSSNumberConstant.h
Samurai_CSSNumberConstant.m
Samurai_CSSNumberDeg.h
Samurai_CSSNumberDeg.m
Samurai_CSSNumberDpcm.h
Samurai_CSSNumberDpcm.m
Samurai_CSSNumberDpi.h
Samurai_CSSNumberDpi.m
Samurai_CSSNumberDppx.h
Samurai_CSSNumberDppx.m

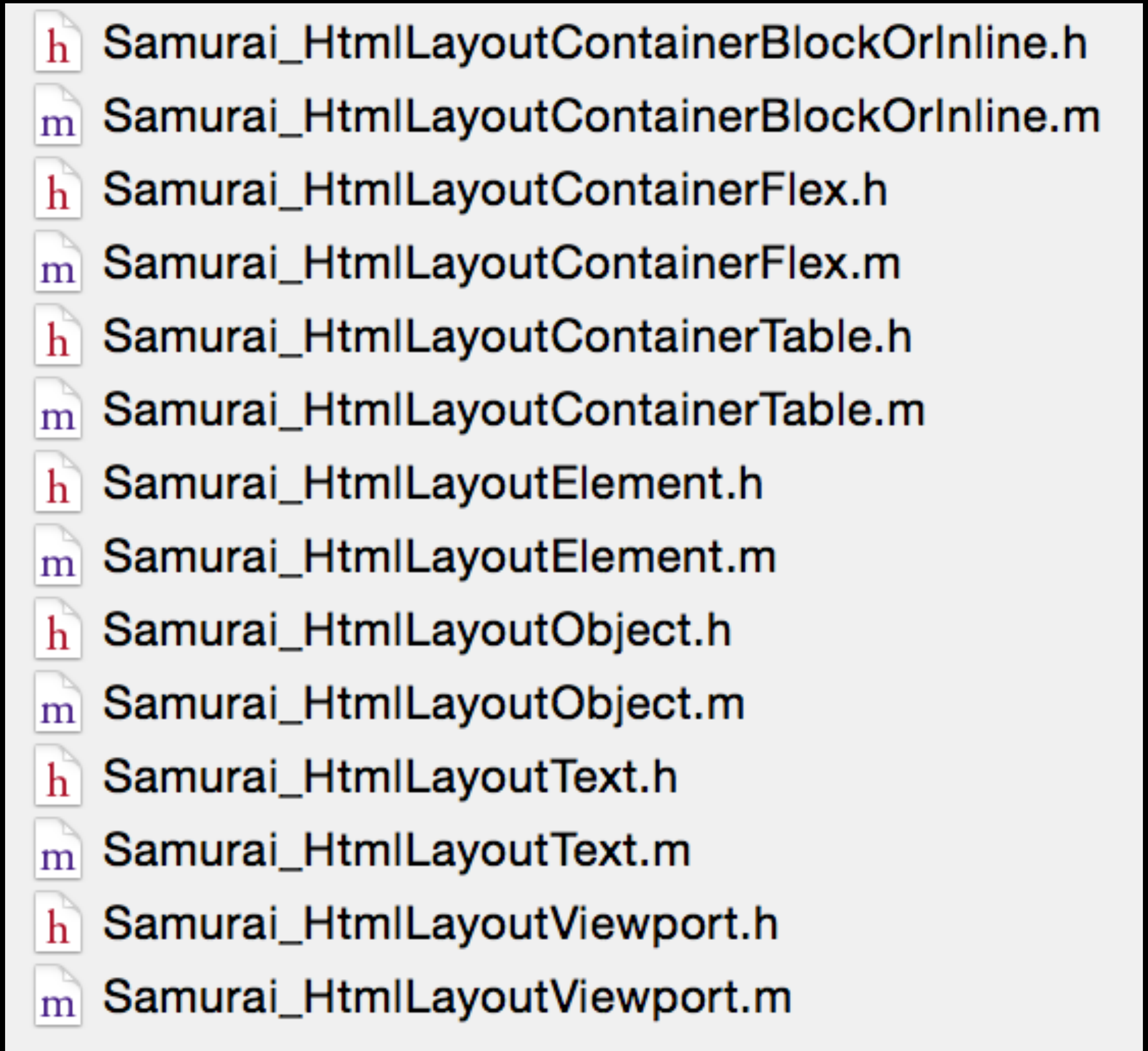


Samurai_CSSNumberEm.h
Samurai_CSSNumberEm.m
Samurai_CSSNumberEx.h
Samurai_CSSNumberEx.m
Samurai_CSSNumberFr.h
Samurai_CSSNumberFr.m
Samurai_CSSNumberGRad.h
Samurai_CSSNumberGRad.m
Samurai_CSSNumberHz.h
Samurai_CSSNumberHz.m
Samurai_CSSNumberIn.h
Samurai_CSSNumberIn.m
Samurai_CSSNumberKhz.h
Samurai_CSSNumberKhz.m
Samurai_CSSNumberMm.h
Samurai_CSSNumberMm.m
Samurai_CSSNumberMs.h
Samurai_CSSNumberMs.m
Samurai_CSSNumberPc.h
Samurai_CSSNumberPc.m
Samurai_CSSNumberPercentage.h
Samurai_CSSNumberPercentage.m



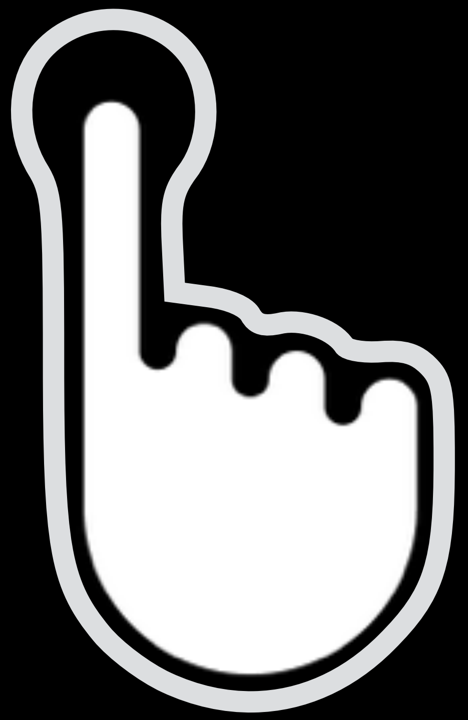
Samurai_CSSNumberPt.h
Samurai_CSSNumberPt.m
Samurai_CSSNumberPx.h
Samurai_CSSNumberPx.m
Samurai_CSSNumberQem.h
Samurai_CSSNumberQem.m
Samurai_CSSNumberRad.h
Samurai_CSSNumberRad.m
Samurai_CSSNumberRems.h
Samurai_CSSNumberRems.m
Samurai_CSSNumberS.h
Samurai_CSSNumberS.m
Samurai_CSSNumberTurn.h
Samurai_CSSNumberTurn.m
Samurai_CSSNumberVh.h
Samurai_CSSNumberVh.m
Samurai_CSSNumberVmax.h
Samurai_CSSNumberVmax.m
Samurai_CSSNumberVmin.h
Samurai_CSSNumberVmin.m
Samurai_CSSNumberVw.h
Samurai_CSSNumberVw.m

Layout support

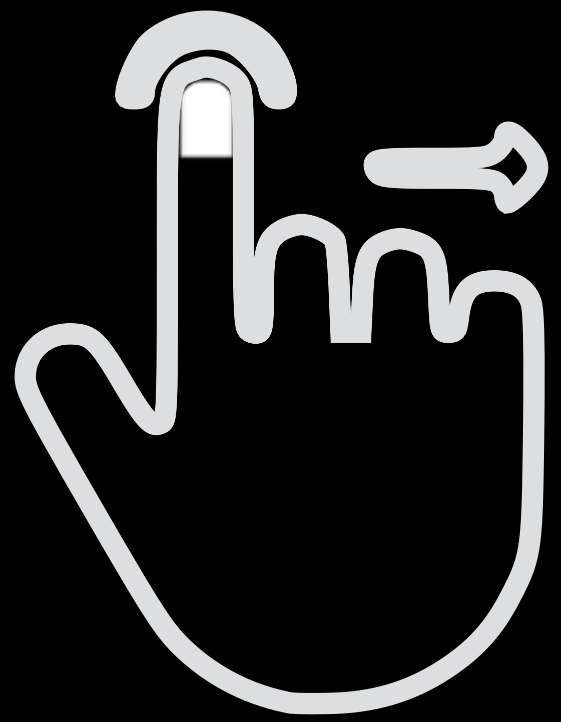


Samurai_HtmlLayoutContainerBlockOrInline.h
Samurai_HtmlLayoutContainerBlockOrInline.m
Samurai_HtmlLayoutContainerFlex.h
Samurai_HtmlLayoutContainerFlex.m
Samurai_HtmlLayoutContainerTable.h
Samurai_HtmlLayoutContainerTable.m
Samurai_HtmlLayoutElement.h
Samurai_HtmlLayoutElement.m
Samurai_HtmlLayoutObject.h
Samurai_HtmlLayoutObject.m
Samurai_HtmlLayoutText.h
Samurai_HtmlLayoutText.m
Samurai_HtmlLayoutViewport.h
Samurai_HtmlLayoutViewport.m

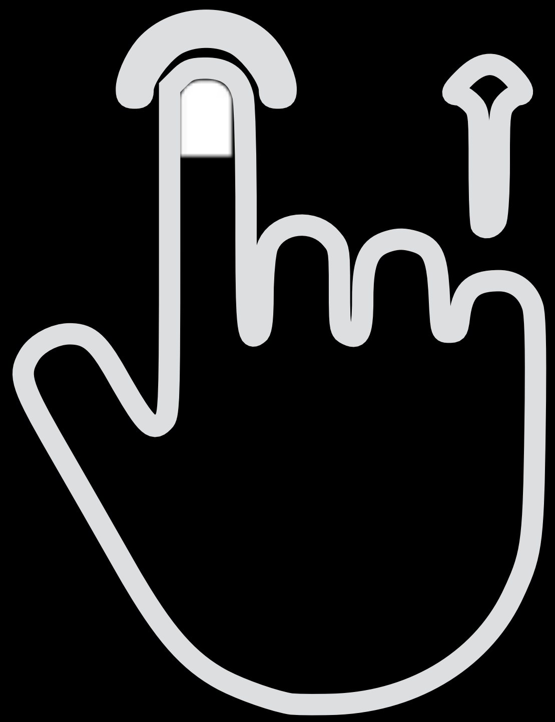
Gesture support



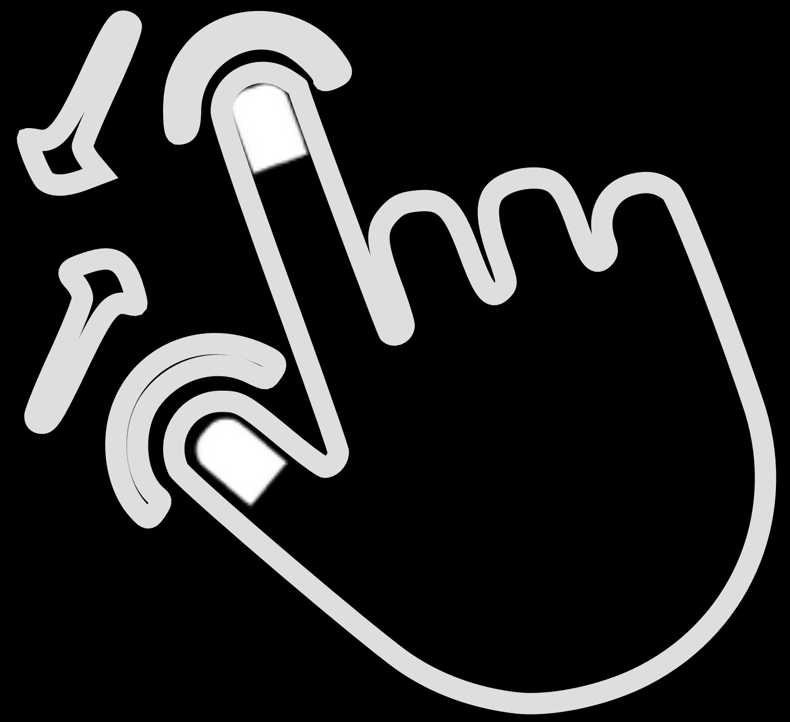
Tap



Swipe



Pan



Pinch

```
onswipe="@selector(doSomething)"
onpinch="@selector(doSomething)"
```

Dynamic template loading

- **File level dynamic loading**
- **Content level hot update**
- Bind to ViewController or View
  - [self **loadTemplate:@“/www/html/index.html”**]
  - [self **loadTemplate:@“http://test.com/index.html”]**

Dynamic data binding

- Bind to ViewController.scope or View.scope
  - Setter
    - self.scope.data = **@[...];**
  - Getter
    - NSDictionary * data = **self.scope.data;**

Vendor library

- gumbo
  - HTML parser, by **@Google**
- katana
  - CSS3 Parser, by **QFish@Geek-Zoo**

Custom default style

• html.css	UA stylesheet, copy from WebKit
• html+native.css	UIKit components stylesheet
• html+samurai.css	UIKit view hierarchy stylesheet

Custom view hierarchy & class

- html+samurai.css
  - **-samurai-view-hierarchy**
    - **hidden, tree, leaf, branch**
  - **-samurai-view-class**
    - Class of native view, **e.g. “UIView”**

```
<html>  tree
    <header>  hidden
    </header>
    <body>  tree
        <UITableView>  leaf
            <UITableViewCell/>  tree
        </UITableView>
        <MyView/>
    </body>
</html>
```

Custom view hierarchy & class

- MyView
  - **- (void) applyDOM:(SamuraiDomNode *)dom**
    - Parse DOM attributes
  - **- (void) applyRender:(SamuraiRenderObject *)render**
    - Create subview, Apply computedStyle, …

Custom view tag

- <**MyView**/>
- @implementation **MyView**
  - - (CGSize) **computeSizeBySize** / ByWidth / ByHeight
  - - (void) **applyDom**:(SamuraiDomNode *)dom
  - - (void) **applyRender**:(SamuraiRenderNode *)render
- **<link href=“my.css”>**
  - **-samurai-view-hierarchy**
  - **-samurai-view-class**

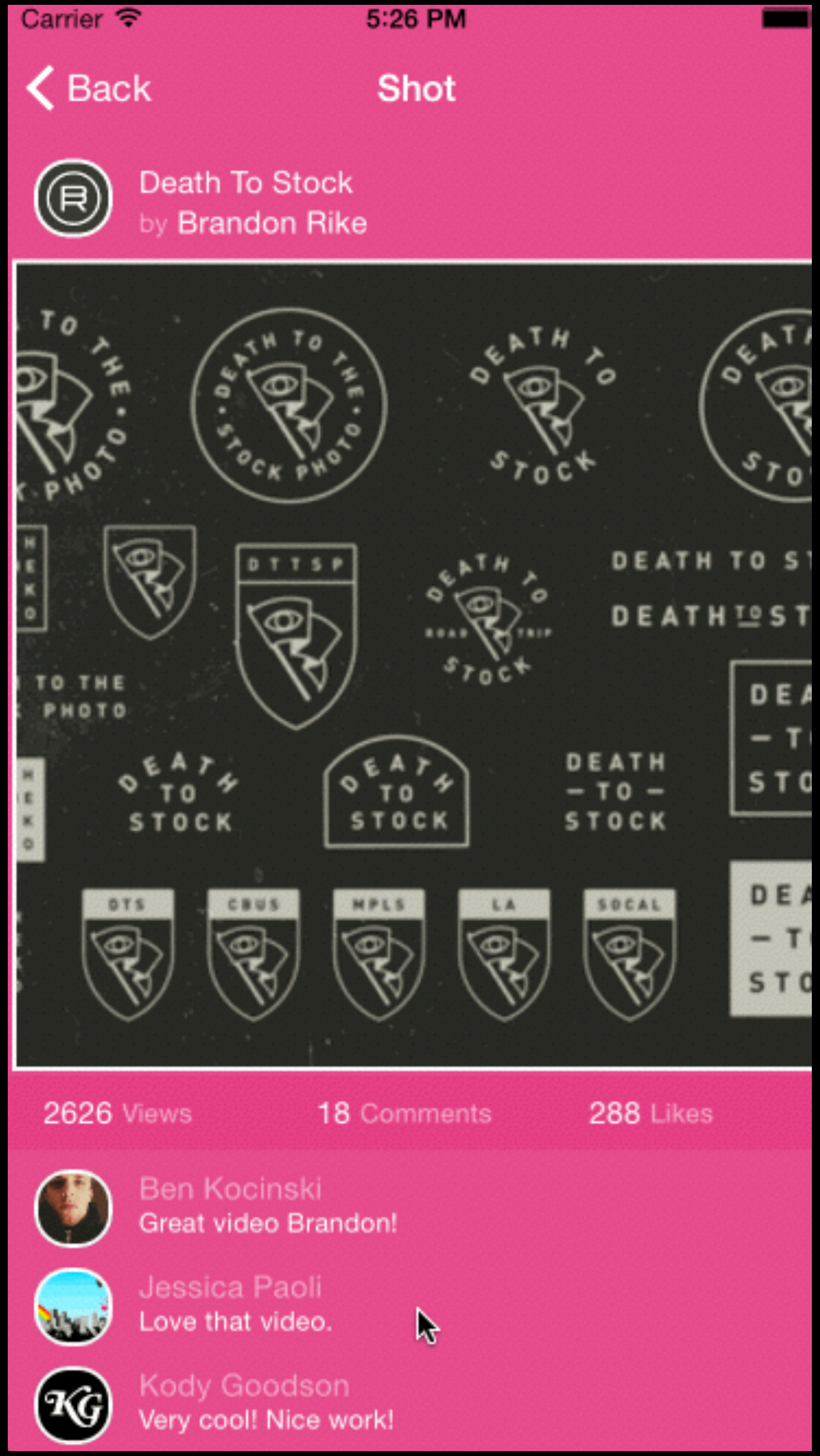
Optimization

- CSSValue cache
  - String > Color, Units
- StyleSheet cache
  - String > StyleSheet
- DOM cache
  - Immutable DOM tree
- Layout cache
  - Fixed width / height

Performance

- Scrolling FPS
    - **> 45**
- View complexity
    - **< 10**

dribbble
Change them
DON'T YOU THINK THAT IF I WERE WRONG, I'D KNOW IT?



Death To Stock
by Brandon Rike

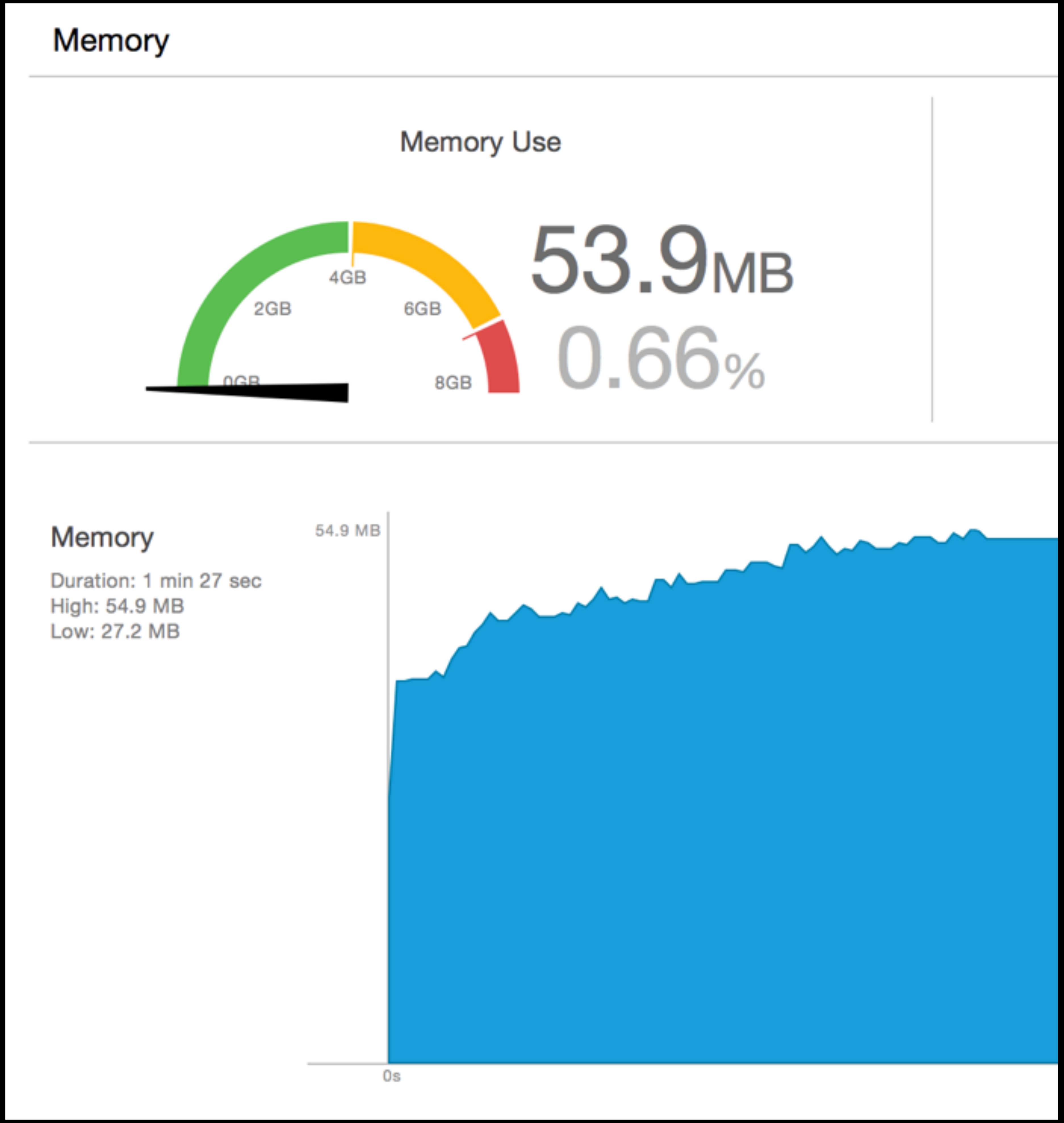
2626 Views 18 Comments 288 Likes

Ben Kocinski
Great video Brandon!

Jessica Paoli
Love that video.

Kody Goodson
Very cool! Nice work!

Memory usage



Memory

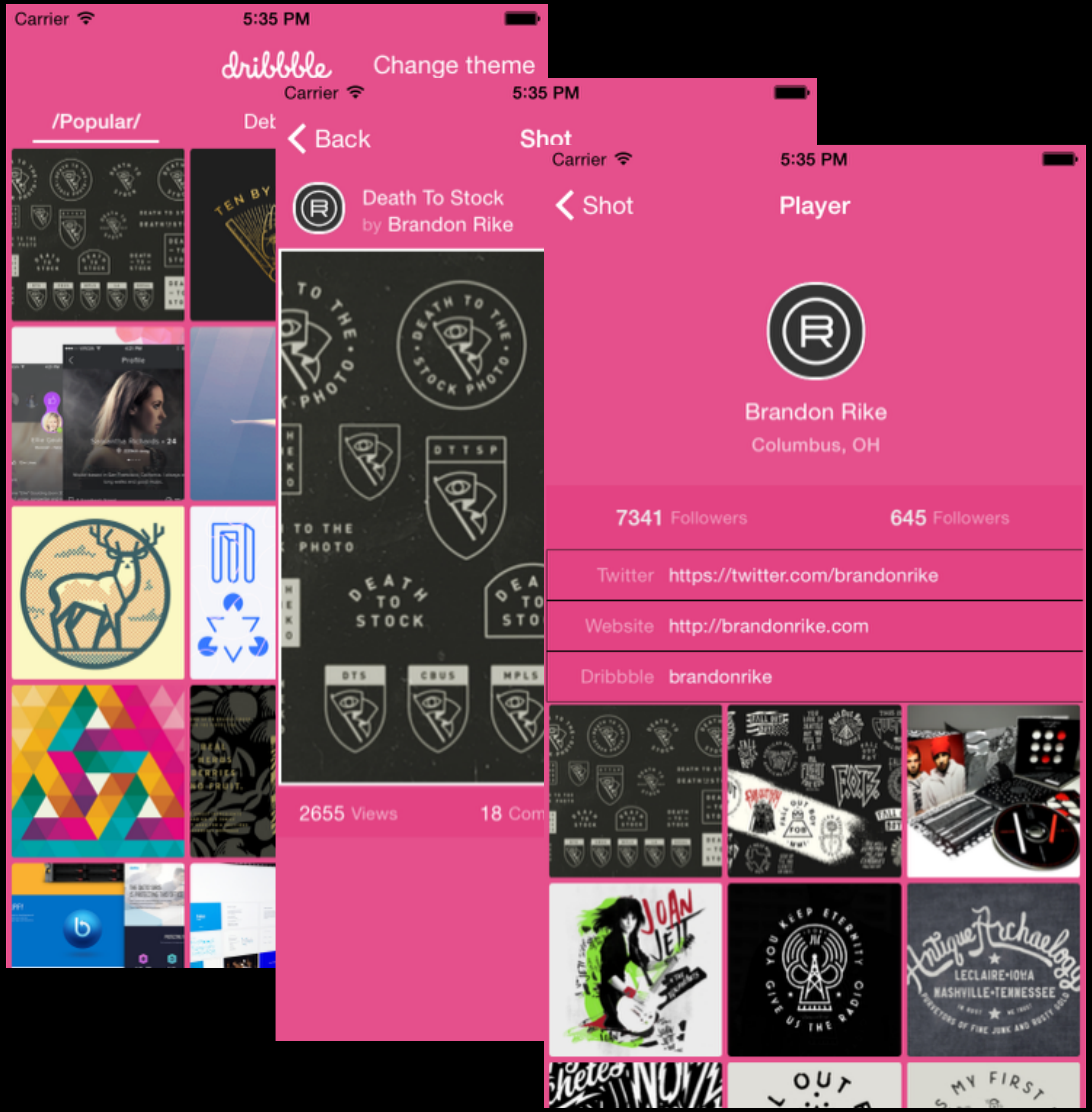
Memory Use

53.9MB

0.66%

Memory

Duration: 1 min 27 sec
High: 54.9 MB
Low: 27.2 MB



Death To Stock
by Brandon Rike

2655 Views
18 Com

Brandon Rike
Columbus, OH

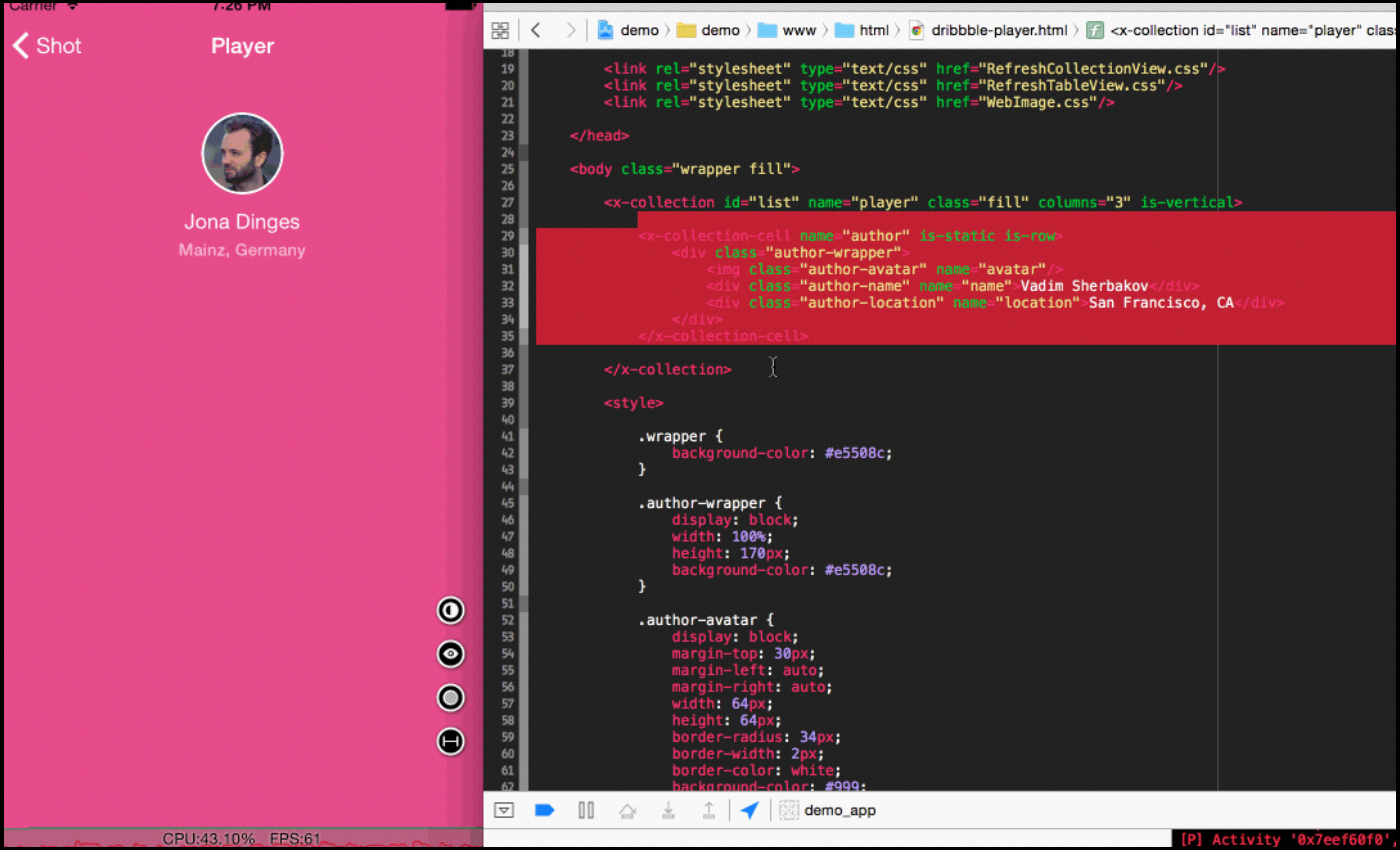
7341 Followers
645 Followers

Twitter https://twitter.com/brandonrike

Website http://brandonrike.com

Dribbble brandonrike

Live-Reload



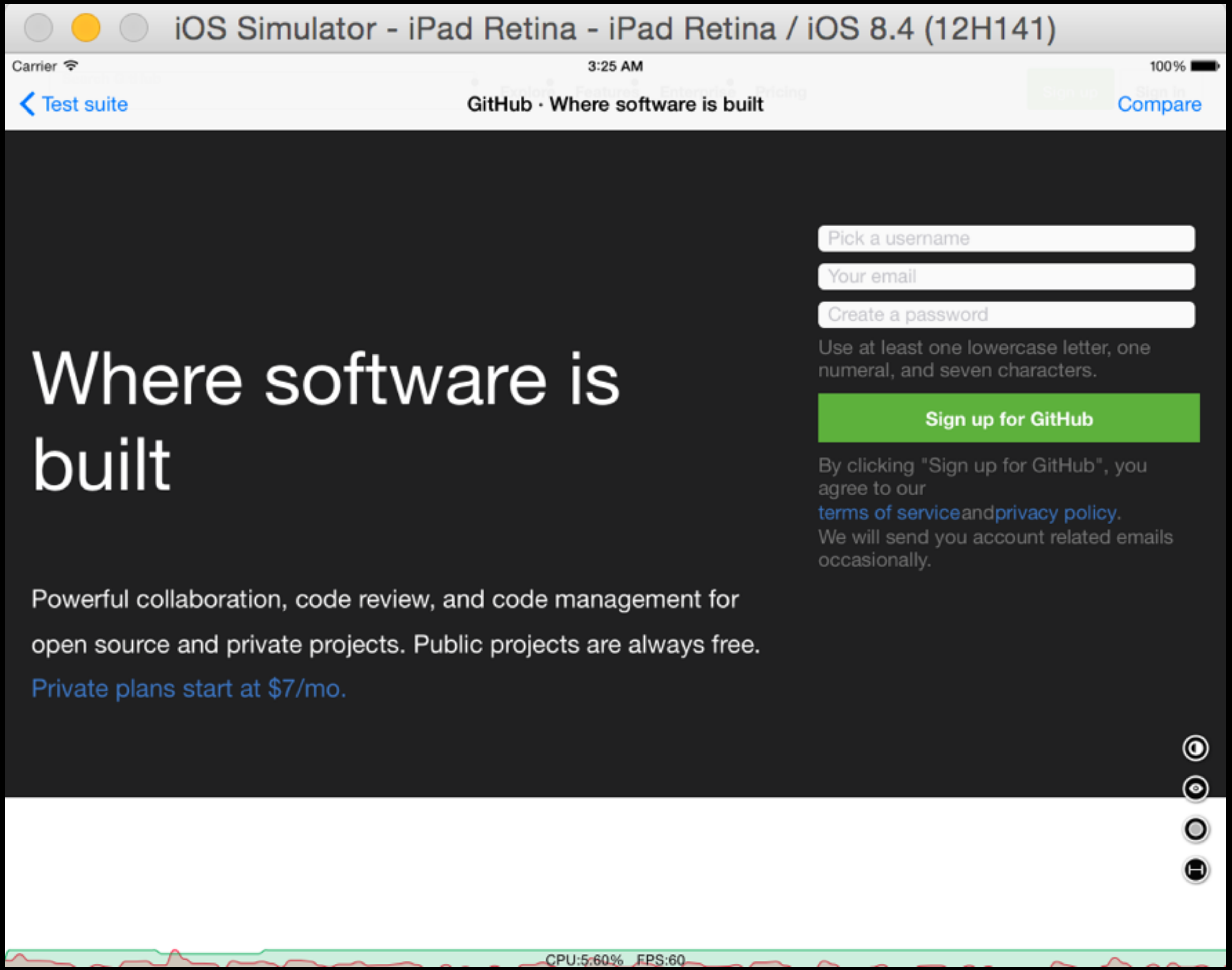
```
        <link rel="stylesheet" type="text/css" href="RefreshCollectionView.css"/>
        <link rel="stylesheet" type="text/css" href="RefreshTableView.css"/>
        <link rel="stylesheet" type="text/css" href="WebImage.css"/>

    </head>

    <body class="wrapper fill">

        <x-collection id="list" name="player" class="fill" columns="3" is-vertical>

            <x-collection-cell name="author" is-static is-row>
                <div class="author-wrapper">
                    <img class="author-avatar" name="avatar"/>
                    <div class="author-name" name="name">Vadim Sherbakov</div>
                    <div class="author-location" name="location">San Francisco, CA</div>
                </div>
            </x-collection-cell>

        </x-collection>

        <style>

            .wrapper {
                background-color: #e5508c;
            }

            .author-wrapper {
                display: block;
                width: 100%;
                height: 170px;
                background-color: #e5508c;
            }

            .author-avatar {
                display: block;
                margin-top: 30px;
                margin-left: auto;
                margin-right: auto;
                width: 64px;
                height: 64px;
                border-radius: 34px;
                border-width: 2px;
                border-color: white;
                background-color: #999;
```

```
[P] Activity '0x7eef60f0',
```

Debugging

- <div debug-dom>
- <div debug-css>
- <div debug-layout>
- <div debug-style>
    - **asm("int3");**

ACID1 - 95%



iOS Simulator - iPad Retina - iPad Retina / iOS 8.4 (12H141)

Carrier 3:25 AM 100%

GitHub · Where software is built

Pick a username

Your email

Create a password

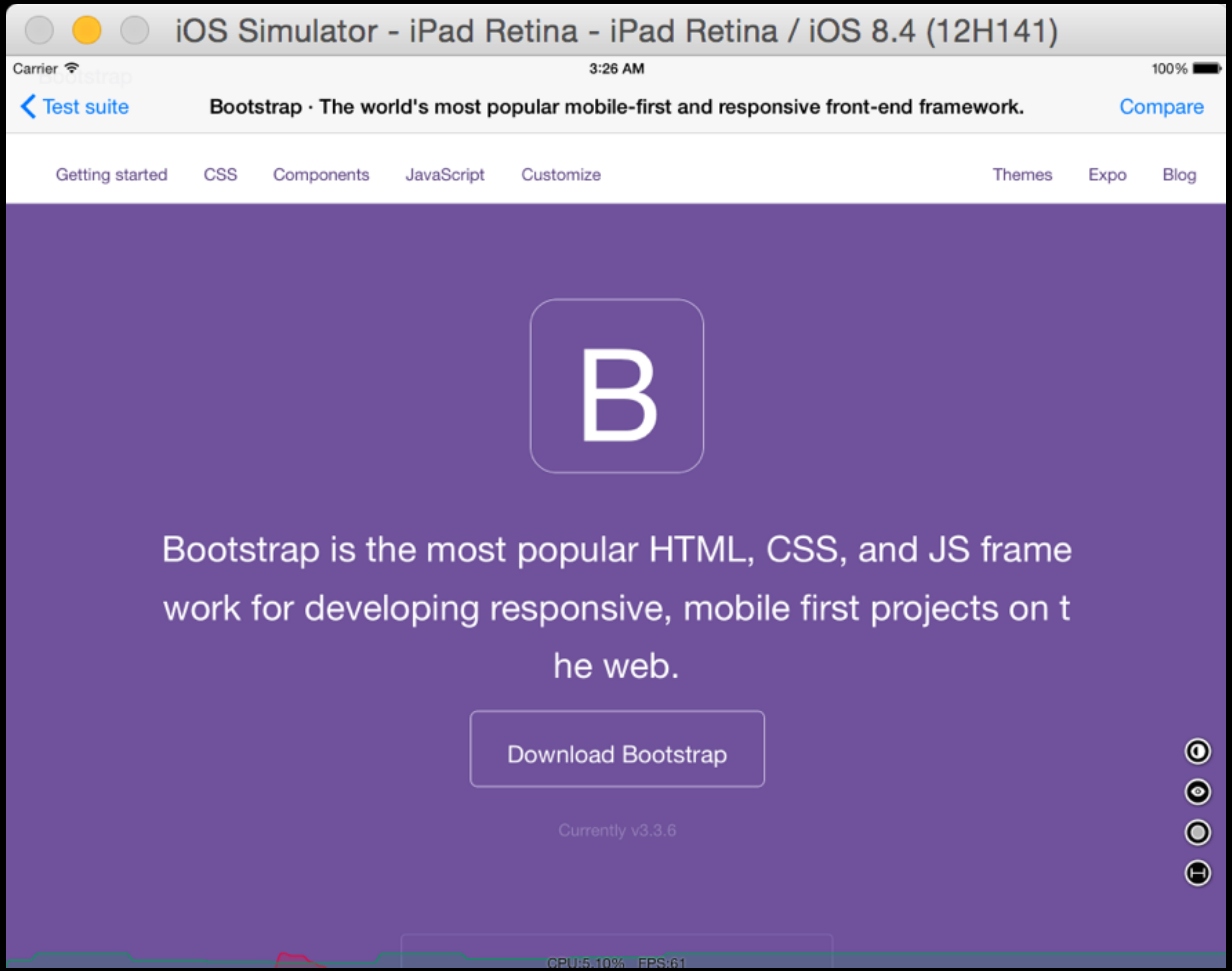
Use at least one lowercase letter, one numeral, and seven characters.

By clicking "Sign up for GitHub", you agree to our terms of serviceandprivacy policy. We will send you account related emails occasionally.

Where software is built

Powerful collaboration, code review, and code management for open source and private projects. Public projects are always free. Private plans start at $7/mo.

CPU:5.60% FPS:60



iOS Simulator - iPad Retina - iPad Retina / iOS 8.4 (12H141)

Carrier 3:26 AM 100%

Bootstrap · The world's most popular mobile-first and responsive front-end framework.

Bootstrap is the most popular HTML, CSS, and JS frame work for developing responsive, mobile first projects on t he web.

Currently v3.3.6

CPU:5.10% FPS:61

ACID2 - 20%

… still hard working on it …

Github search ‘samurai-native’

gavinkwoe@github / qfish@github

Fin.

Presented by Geek-Zoo Studio 2015 @ D2Conf Hangzhou