Android, iOS and Hybrid Applications

Mobile-Development

App Bewertung und Ablauf

- Es wird insgesamt 3 Noten geben:
  - Projektwoche - technische Umsetzung der App
  - Einzelarbeit - technische Umsetzung der App
  - Einzelarbeit - Präsentation der App

App Bewertung und Ablauf

- Technische Bewertung:
  - Siehe separates Excel für Details (github Doku-Repo)

App Bewertung und Ablauf

- Technische Bewertung:
  - Architektur
  - Implementation
  - Anforderungen
  - Lauffähig
  - Benutzeroberfläche
  - Dokumentation
  - Tests

App Bewertung und Ablauf

- Präsentation Bewertung:
  - Einführung
  - Logischer Aufbau
  - Demonstration Funktionen
  - Zeit (10 Minuten)

App Bewertung und Ablauf

- Termine Abgabe - 29.06.2020
    - Letzte Fragerunde
    - Nachträgliche Änderungen für Präsentation erlaubt
    - Pull wird um 21:30 gemacht für die Bewertung
    - Bewertung wird per Mail vorab zugeschickt

App Bewertung und Ablauf

- Termine Präsentation - 06.07.2020
  - Lorenzo: 17:50-18:10
  - Omer: 18:15-18:35
  - Tobias: 18:40-19:00
  - Micha: 19:05-19:25

App Bewertung und Ablauf

- Termine Präsentation - 06.07.2020
  - Emmanuel: 19:30-19:50
  - Raphael: 19:55-20:15
  - Dietrich: 20:20-20:40
  - Marco: 20:45-21:05

- Termine können untereinander abgetauscht werden.

App Bewertung und Ablauf

- Fragen?

Overview

- Hybrid Applications
- Interoperability with the native part
  - Design a possible interface
  - Present your approach
- Create a small working sample
- (Introduction to modern Web-Development)

Hybrid Applications

- Native Part which provides a JS-Interface
- More then 50% market share (hard to prove)
- Browsers support HTML5/CSS3 and ES6
  - https://caniuse.com/

Hybrid Applications

- Pros
  - Share or reuse (UI)-Code (from website etc.)
  - It’s easier to find Web-Devs then native Devs
  - Possible to update without going trough the store
  - Fallbacks/Combination with native possible

Hybrid Applications

- Cons
  - Sometimes don't feel that "responsive"
  - Getting better with later releases/engines
- You need to understand both worlds (native & web)

WebViews

- iOS
  - WKWebView
  - **Don't** use UIWebView
- Android
  - WebView
  - Updates independent of OS (since 4.4.4/19)

https://developer.apple.com/documentation/webkit/wkwebview

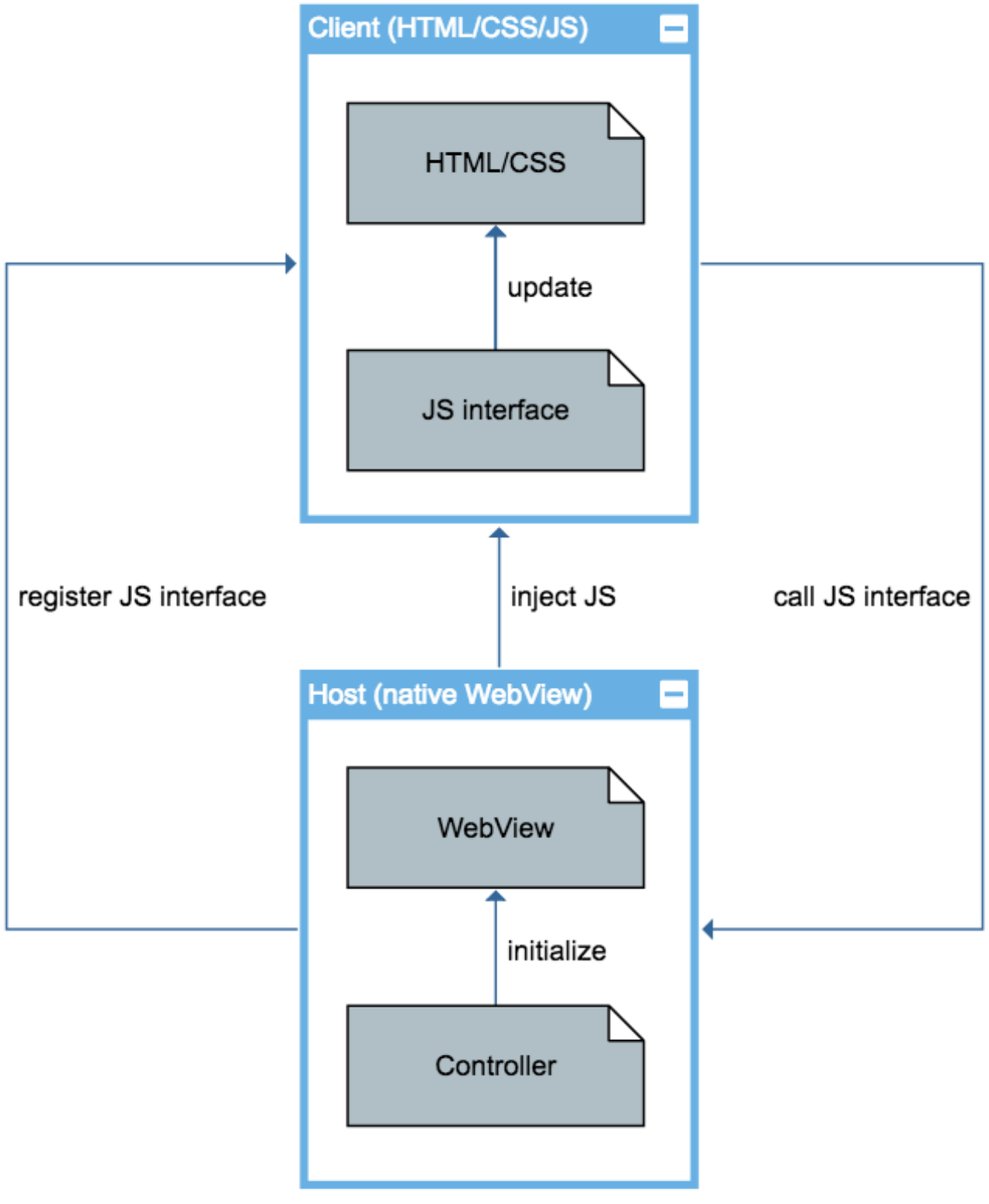
https://developer.android.com/reference/android/webkit/WebView

WebViews

- The control is a wrapper - they run in their own process
- You're not limited to only use one WebView
- Load local HTML pages or remote ones
- Think about CORS when using a mix

https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

ARCHITECTURE



Setup the App

- Clone the sample
  - https://github.com/FabrizioNiedda/webview-example

- It’s a simple Android app (Xamarin)

Register a JS Interface

```
// Set the content layout which contains a simple web view.
SetContentView(Resource.Layout.activity_main);

// Extract the web view from the layout.
var webView = (WebView)FindViewById(Resource.Id.webView);

// Configure WebView to allow JS and inject our custom interface.
webView.Settings.JavaScriptEnabled = true;
webView.AddJavascriptInterface(new JavaScriptInject(this), "Native");

// Load a local HTML file.
webView.LoadUrl("file:///android_asset/index.html");
```

https://docs.microsoft.com/en-gb/dotnet/api/android.webkit.webview.addjavascriptinterface#Android_Webkit_WebView_AddJavascriptInterface_Java_Lang_Object_System_Strin

Register a JS Interface

```
public class JavaScriptInject : Object
{
  /// <summary>
  /// Annotate methods with the <see cref="JavascriptInterfaceAttribute"/> and
  /// the <see cref="ExportAttribute"/> to call them from JS.
  /// </summary>
  [JavascriptInterface]
  [Export("doSomething")]
  public void FromJavaScript()
  {
  }

  /// <summary>
  /// Annotated methods can also accept parameters.
  /// </summary>
  [JavascriptInterface]
  [Export("doSomething")]
  public void FromJavaScript(string message)
  {
  }
}
```

Invoke Native from JS (WebView -> Native)

- Invoke it with Native.yourMethod()

```
<input type="button" onclick="Native.doSomething()" value="Invoke native" />
```

- Also possible with parameters

```
<input type="button" onclick="Native.doSomething('Another message...')" value="Invoke native with param" />
```

Inject JS (Native -> WebView)

```
webView.EvaluateJavascript("do some JS magic...", null);
```

```
webView.EvaluateJavascript("do some JS magic...", new Callback());
public class Callback : Java.Lang.Object, IValueCallback
{
  public void OnReceiveValue(Object value)
  {
    // Do something with the value...
  }
}
```

https://docs.microsoft.com/en-gb/dotnet/api/android.webkit.webview.evaluatejavascript

Example

- Walkthrough

Exercise

- Group up
- Setup the basic Android project
- Clone the Repo for your group
- Think about an approach on how to create a messaging bus between Native and Web
- Present your solution/idea