BUILD MOBILE APPS USING APACHE FLEX

(YES, INCLUDING IOS!)

ApacheCon Core 2015 OmPrakash Muppirala

About Me

- OmPrakash Muppirala
- Lead UI Developer at Machine Zone
- Member@ASF
- Apache Flex Committer and PMC Member
- UI Architect with 15+ years experience building UI for enterprise and consumer apps
- @bigosmallm

Apache Flex for Mobile – Why?

- Adobe AIR
- Cross platform, maximum code reuse
- Ease of development
- No context switching between platforms
- FXG, Spark Skinning
- AS3.0, OOP, familiar design patterns and micro-frameworks
- Multi resolution targeting
- New developments at Apache Flex for Mobile apps

MOMENTUM BEHIND ADOBE AIR

- Adobe AIR v19.0 released on SEP-21-2015
- More than 100,000 unique applications across Desktop, iOS and Android
- ActionScript Workers for Android and iOS
- Ever-growing list of Adobe Native Extensions – for things that AIR cannot do today
  - Check out http://www.riaxe.com/blog/200-adobe-air-anes/ (Free and Paid)

Apache Flex Mobile - Latest Developments

- New Android and iOS skins
- New components/features (Mobile Grid, Scrolling StageText)
- New Tour de Mobile Flex app
  - https://github.com/apache/flex-examples/tree/develop/TourDeFlexMobile
- iOS status bar fix
- Support for low res (120DPI) and hi-res (480, 640 DPI) devices
- CSS-Media queries for OS version
- CSS-Media queries for device dimensions
- And of course, all the Flex SDK bug fixes and performance improvements

Apache Flex Showcase

- Check out all the cool apps built with Apache Flex
  - http://flex.apache.org/community-showcase.html
  - Includes a whole bunch of apps deployed to the Apple App Store and Google Play Store
  - If you have build a great Apache Flex app, mobile or otherwise, please consider adding it to the showcase!

BEST PRACTICES

LETS LOOK AT SOME GOOD PRACTICES WHILE CREATING CROSS PLATFORM APPLICATIONS

Setting your projects up

- Target Web, Desktop, iOS, Android in one codebase
- Core Flex Library project contains almost all the code
- Web, Desktop and Mobile projects simply contain app container logic and respective style declarations
- Use different CSS files to drive visual look and feel for iOS vs. Android
- Use Capabilities.xxx to build device specific experiences
- Use xxx.isSupported to build feature specific experiences (ex. Accelormeter.isSupported, Camera.isSupported, etc.)

SETTING YOUR PROJECTS UP

HERE IS HOW AN IDEAL FLASH BUILDER PROJECT SETUP LOOKS LIKE

HTTPS://GITHUB.COM/BIGOSMALLM/TWITTERTRENDS

CREATING A NEW SKIN

MOBILE SKINNING

- Separate components' view from logic
- Use CSS to set styles
- In CSS, use CSS Media Queries to set styles and skins based on DPI, OS type, device-width, device-height, OS version.
- In AS skin code, use applicationDPI to create visuals per DPI
- All mobile skins are based off of the MobileSkin class
- The skin class usually references FXG assets
- Pay attention to the methods drawBackground(), commitCurrentState() and layoutContents() This is where most of the action happens!
- Most times, copying existing skin classes and using new FXG assets is the best approach

FXG Assets – Vectors Rule!

- FXG skins for resolution independent vector graphics
  - Rect, Ellipse, Fill, Strokes, etc. instead of bitmap skins
- Build your visuals in Illustrator or Flash Pro
- Export to FXG format
- Convert them into readily usable FXG assets using the FXGTools project
  - https://github.com/apache/flex-utilities/tree/master/FXGTools
  - Splits up individual FXG files, removes unnecessary Groups, removes unnecessary namespaces
- For all DPIs, use same FXG assets. They scale up/down as needed
- Use scaleGridLeft, scaleGridRight, scaleGridTop, scaleGridBottom for perfect scaling

DEMOS

QUESTIONS?