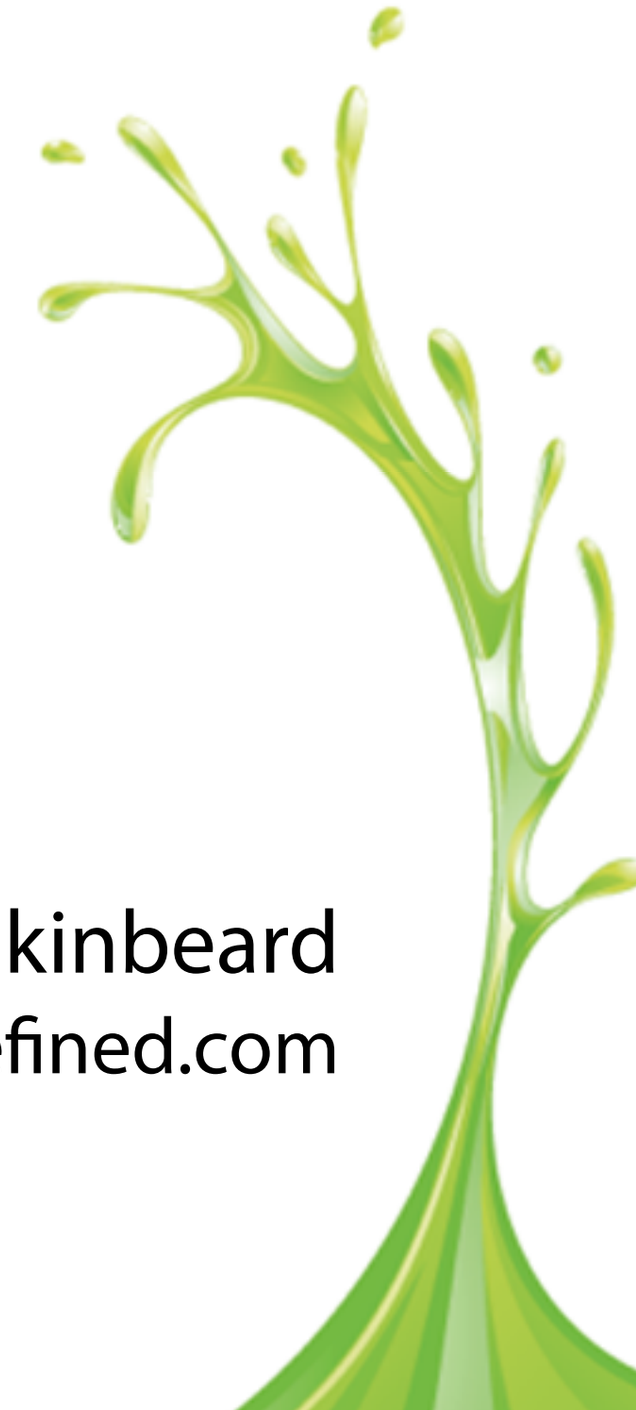
UNIVERSAL MIND

Concepts and Strategies for Creating Reusable Components



Ben Clinkinbeard
http://www.returnundefined.com

UNIVERSAL MIND

What are we going to cover?

- Two “levels” of reusability
- Weigh the benefits and use good judgment
- Components should be:
  - easy to utilize
  - reusable
  - customizable
  - style friendly
  - useful (not just usable)

UNIVERSAL MIND

Why?

UNIVERSAL MIND

You

- **Good:** Creating new components and applications

- **Bad:** Supporting, editing or copy/pasting old components

UNIVERSAL MIND

People using your component

- **Good:** Your component just works
    - Discovery of depth and elegance can/should happen later

- **Bad:** It takes 15 minutes of reading docs and code to get up and running

UNIVERSAL MIND

OK smart guy, how?

UNIVERSAL MIND

flexmdi

- Project site - *http://code.google.com/p/flexlib/*
- Brian Holmes - *http://brianjoseph31.typepad.com/smashedapples/*
- Brendan Meutzner - *http://www.meutzner.com/blog/*

- Conceived at 360|Flex Seattle, August 2007
- Released September 2007

UNIVERSAL MIND

Two “levels” of reusability

- Loosely coupled

- Polished, fit for cataloguing/distribution

UNIVERSAL MIND

Loosely coupled components

- Do it every time
- Use and bind to local (but public) vars
- Data set by ancestors
- Self-contained

UNIVERSAL MIND

Loosely coupled example

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">
   <mx:Script>
      <![CDATA[

         [Bindable]
         public var someVar:String;

      ]]>
   </mx:Script>

   <mx:Label text="{ someVar }" />
</mx:Canvas>
```

UNIVERSAL MIND

Polished components

- Do it when you can
- Harder
- More time consuming (by far)
- Think like end users

UNIVERSAL MIND

Composition over “in there”-itance

- Division of responsibilities
    - Classic tenet of OOP and common refactoring tactic
- One component != one class
- Expose modularity / assignment
    - MDIWindowControlsContainer
    - flexmdi effects
- Inversion of control / Dependency Injection

UNIVERSAL MIND

Hide the details

- Another core principle of OOP - Encapsulation
- Less exposure means more freedom to change but...
- Don’t be stingy
- Automate tedious tasks
    - MDICanvas
    - mdiManager.windowEventProxy()

UNIVERSAL MIND

Providing default behaviors (your job)

- Listen for own events
- Low priority listener to ensure late/last execution
- EventPriority.DEFAULT_HANDLER:int = -50
    - addEventListener(type:String, listener:Function, useCapture:Boolean = false, **priority:int = 0,** useWeakReference:Boolean = false);
- Events must be cancelable
    - Event(type:String, bubbles:Boolean = false, **cancelable:Boolean = false**);

UNIVERSAL MIND

Modifying default behaviors (their job)

- "Normal" listeners will get called first
  - mdiWindow.addEventListener(MDIManager.WINDOW_CLOSE, onWindowClose);
- Cancel default handler with event.preventDefault()
- Use event.clone() to store copy for later execution

UNIVERSAL MIND

Executing default behaviors (your job)

- Default handler is public for delayed / manual calls
    - mdiManager.executeDefaultBehavior(event);
- Behavior is conditional
    - if(!event.isDefaultPrevented())

UNIVERSAL MIND

Follow conventions

- Use life cycle functions when possible / appropriate
  - MDIWindowControlsContainer
- Styles as styles
- Metadata...

UNIVERSAL MIND

Provide default styles

- Static initializer
    - private static function initializeStyles(){...}
    - private static var stylesInitialized():Boolean = initializeStyles();
- Docs are wrong - use defaultFactory(), not setStyle()
    - MDIWindow.classConstruct();

UNIVERSAL MIND

Metadata (is mandatory)

- Compiler instructions
- Code completion and MXML assignments

- [Event(name=”minimize”, type=”flexmdi.events.MDIWindowEvent”)]
- [Style(name=”closeBtnStyleName”, type=”String”, inherit=”no”)]
- [Bindable]

UNIVERSAL MIND

Make things easy

- Expose things the framework doesn't
    - mdiWindow.getTitleTextField()
    - mdiWindow.getTitleIconObject()
- Proxy properties
    - mdiCanvas.enforceBoundaries
- Provide base implementations
    - MDIEffectsDescriptorBase
- Use flexible classes like LayoutContainer

UNIVERSAL MIND

Thank you!

(tip your waitress)