BEHAVIOURS

IN IOS APPS

KRZYSZTOF ZABŁOCKI @MEROWING_

WHAT IS BEHAVIOUR?

WHAT IS BEHAVIOUR?

- focuses on user interaction
- implements specific role

WHAT BENEFITS DOES USING BEHAVIOURS BRING?

QUALITY & EFFECTIVNESS

QUALITY

- Cleaner code
- Easier to maintain
- Tested
- Shared codebases

QUALITY

Avoiding Massive View Controllers by off-loading functionality into separate small classes.

Small classes are easier to maintain and modify.

QUALITY

Those classes tend not to have dependency on application logic, which means they can be re-used across different applications.

They are also easy to test.

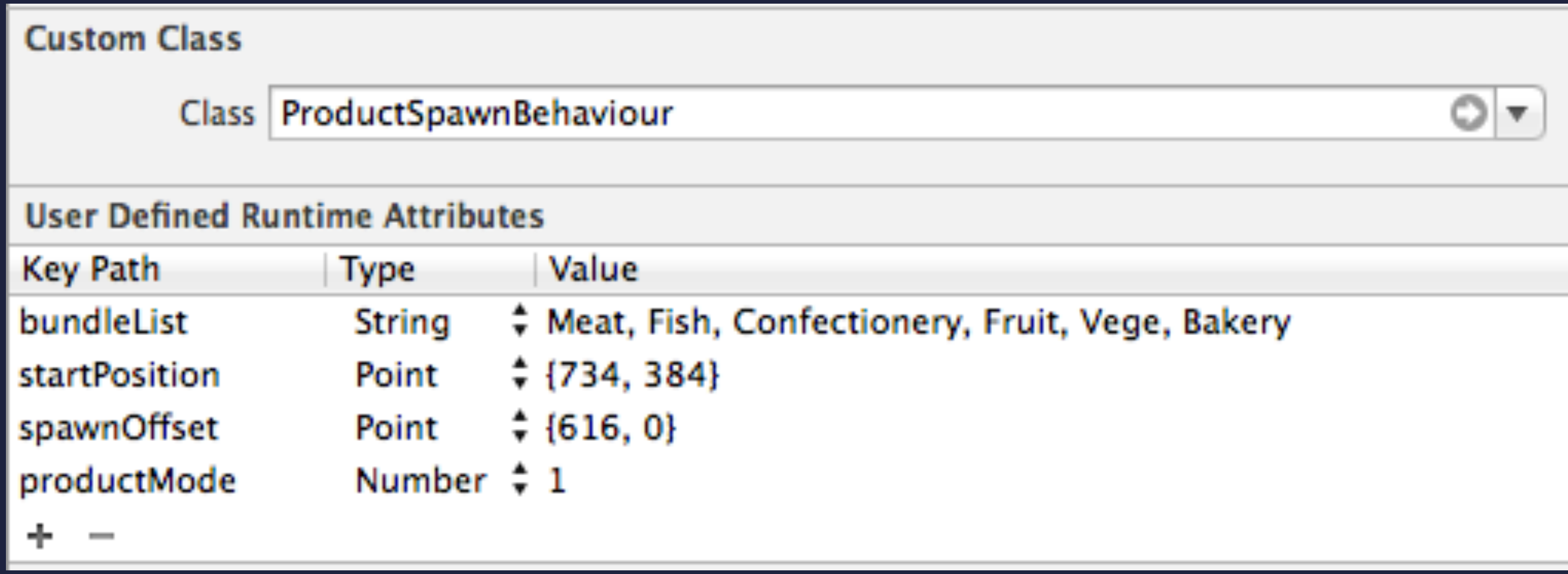
EFFECTIVENESS

- Non-Developers can modify application behaviour
- Designers can tweak variables

You can focus on new features instead of wasting your time tweaking parameters.

BUILDING BEHAVIOURS

RUNTIME ATTRIBUTES



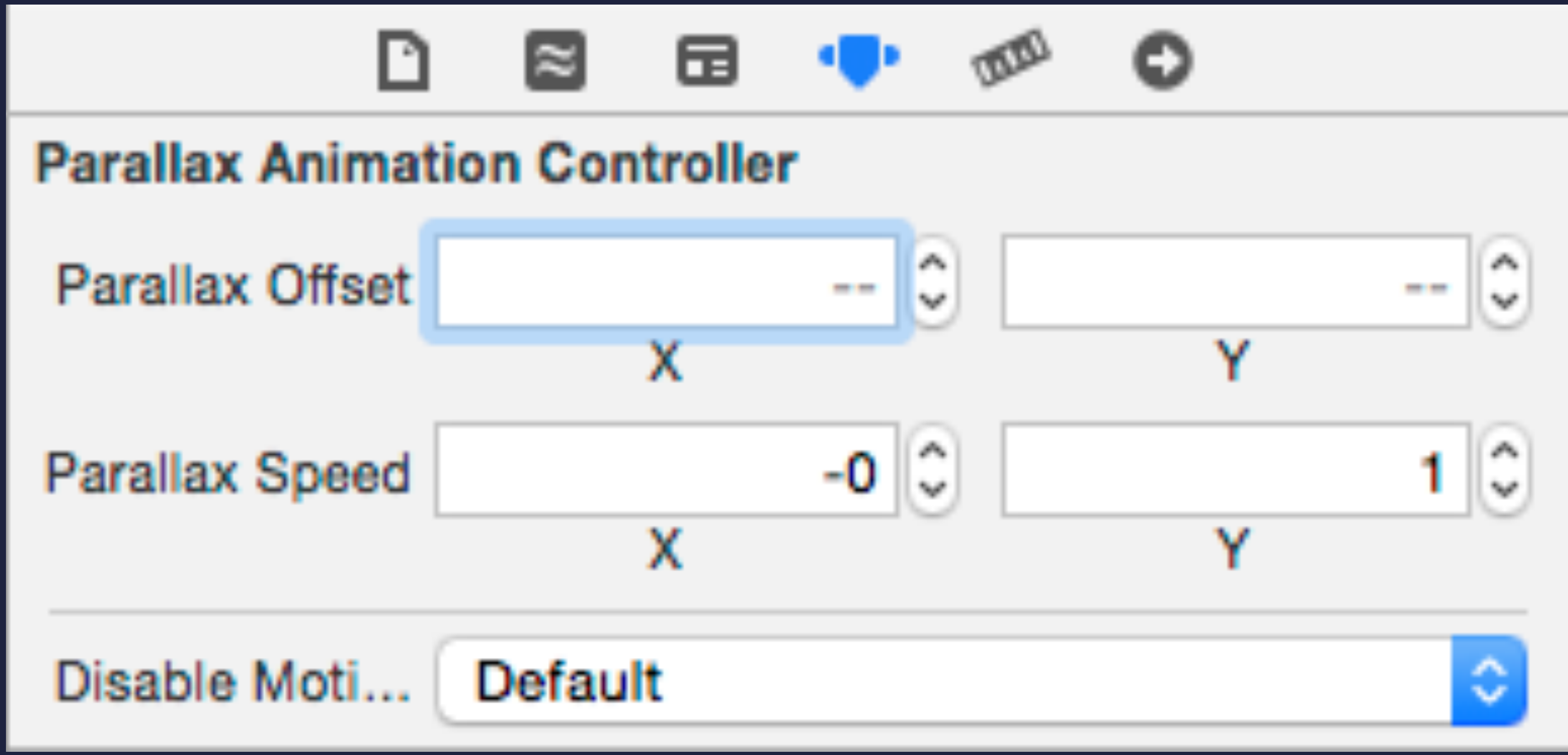
Custom Class

Class: ProductSpawnBehaviour

User Defined Runtime Attributes

Key Path	Type	Value
bundleList	String	Meat, Fish, Confectionery, Fruit, Vege, Bakery
startPosition	Point	{734, 384}
spawnOffset	Point	{616, 0}
productMode	Number	1

INSPECTABLES WITH XCODE 6



Parallax Animation Controller

Parallax Offset -- (X) -- (Y)

Parallax Speed -0 (X) 1 (Y)

Disable Moti... Default

BEHAVIOUR LIFETIME

- Objects created from interface builder are immedietely released if there is no strong reference to them
    - This usually requires adding properties to view controllers

Not ideal because then removing a behaviour also requires removing that property

BEHAVIOUR LIFETIME

We can leverage associated objects to reverse lifetime binding:

- Behaviour will decide how long to keep itself alive
- Removing behaviour or adding new ones will *NOT* require modifying controller code.

```
- (void)bindLifetimeToObject:(id)object
{
  objc_setAssociatedObject(object, (__bridge void *)self, self, OBJC_ASSOCIATION_RETAIN_NONATOMIC);
}

- (void)releaseLifetimeFromObject:(id)object
{
  objc_setAssociatedObject(object, (__bridge void *)self, nil, OBJC_ASSOCIATION_RETAIN_NONATOMIC);
}
```

BEHAVIOUR EVENTS

BEHAVIOUR EVENTS

It's useful to be able inform controller that an event has occurred:
eg. let view controller know that user selected an image

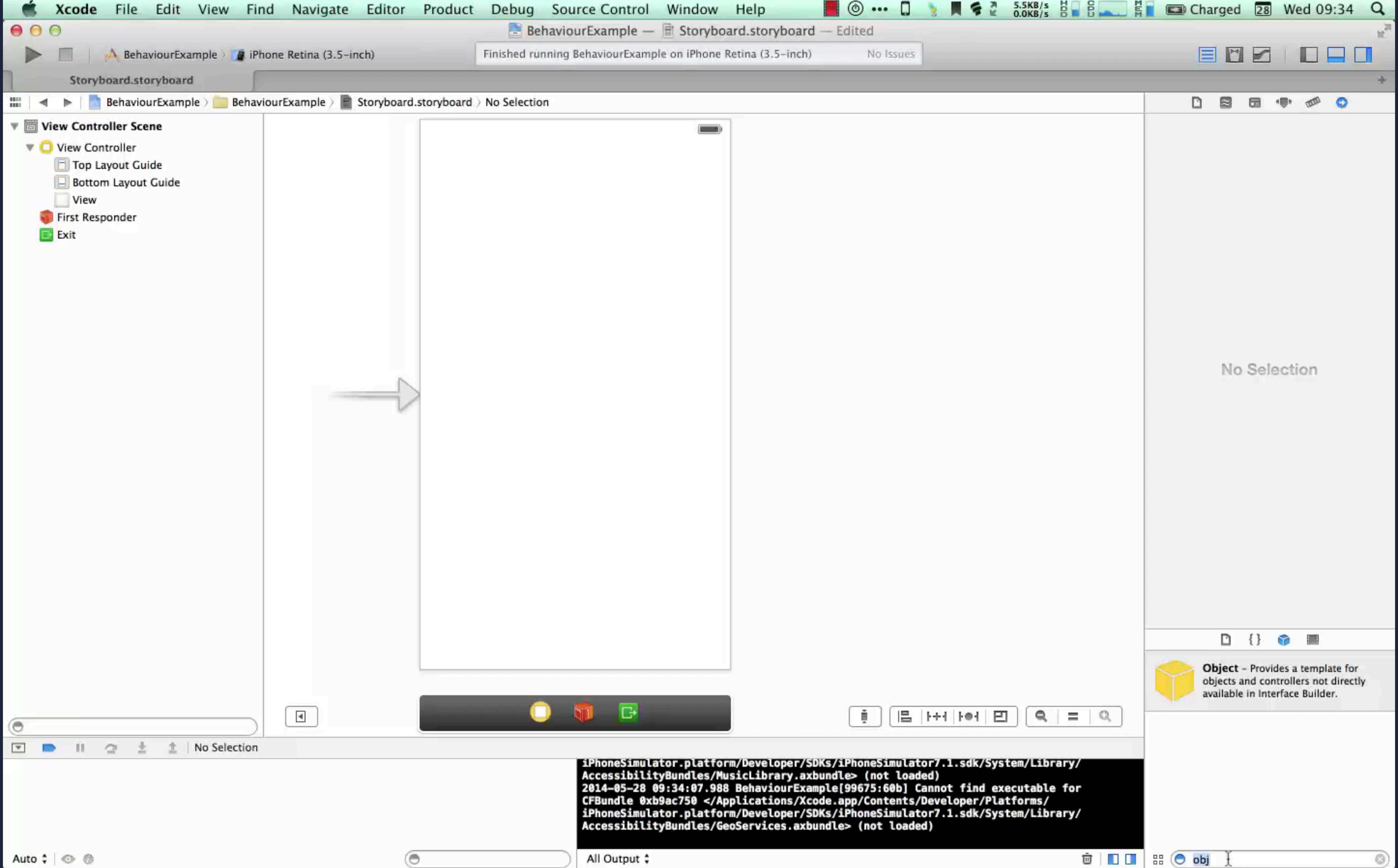
By making a Behaviour subclass of UIControl we are able to leverage iOS target-action pattern.

```
[self sendActionsForControlEvents:UIControlEventValueChanged];
```

SAMPLE BEHAVIOURS

- Animations
- Image picking
- Drag & Drop
- Character limiter (think twitter)

Placeholders
File's Owner
First Responder
View
No Selection


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Charged Wed 09:34
BehaviourExample — Storyboard.storyboard — Edited
BehaviourExample iPhone Retina (3.5-inch)
Storyboard.storyboard
BehaviourExample BehaviourExample Storyboard.storyboard No Selection
View Controller Scene
View Controller
Top Layout Guide
Bottom Layout Guide
View
First Responder
Exit
No Selection
Auto
All Output
obj

Finished running BehaviourExample on iPhone Retina (3.5-inch) No Issues

Object - Provides a template for objects and controllers not directly available in Interface Builder.

```
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.1.sdk/System/Library/
AccessibilityBundles/MusicLibrary.axbundle> (not loaded)
2014-05-28 09:34:07.988 BehaviourExample[99675:60b] Cannot find executable for
CFBundle 0xb9ac750 </Applications/Xcode.app/Contents/Developer/Platforms/
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.1.sdk/System/Library/
AccessibilityBundles/GeoServices.axbundle> (not loaded)
```

CONCLUSION

- Cleaner code
- Reusability
- Ease of changes
- non-coders can help out

THANK YOU

FOLLOW @MEROWING_