Sling Models

Justin Edelson | Technical Architect

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Agenda

- Background & Goals
- Usage
- Extensions

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

- Let’s say you want to adapt a Resource into some domain object…

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

```
public class OldModel {

    private String title;
    private String description;

    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public String getDescription() {
        return description;
    }
    public void setDescription(String description) {
        this.description = description;
    }
}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

```
@Component
@Service
@Properties({
    @Property(name=AdapterFactory.ADAPTABLE_CLASSES, value="org.apache.sling.api.Resource"),
    @Property(name=AdapterFactory.ADAPTER_CLASSES,
        value="com.adobe.people.jedelson.slingmodels.demo.OldModel")
})
public class OldModelAdapterFactory implements AdapterFactory {

    @SuppressWarnings("unchecked")
    public <AdapterType> AdapterType getAdapter(Object adaptable, Class<AdapterType> type) {
        if (adaptable instanceof Resource && type.equals(OldModel.class)) {
            OldModel model = new OldModel();
            ValueMap map = ResourceUtil.getValueMap((Resource) adaptable);
            model.setTitle(map.get(”title", String.class));
            model.setDescription(map.get(”description", String.class));
            return (AdapterType) model;
        } else {
            return null;
        }
    }
}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

```
OldModel myModel = resource.adaptTo(OldModel.class)
```

```
<sling:adaptTo adaptable="${resource}" adaptTo=”…
    OldModel" var=”myModel" />
```

```
<div data-sly-use.myModel =“…OldModel”></div>
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

```
@Model(adaptables = Resource.class)
public class NewModel {

    @Inject
    private String title;

    @Inject
    private String description;

    public String getTitle() {
        return title;
    }
    public String getDescription() {
        return description;
    }

}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

NewModel myModel = resource.adaptTo(NewModel.class)

<sling:adaptTo adaptable="${resource}" adaptTo=”…

NewModel" var=”myModel" />

<div data-sly-use.myModel=“…NewModel”>

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

- The “old” way: 30+ LOC
- The “new” way: 13 LOC
  - Plus one extra bundle header:

```
<Sling-Model-Packages>com.adobe.people.jedelson.slingmodels.demo</Sling-Model-Packages>
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

What is Sling Models?

```
@Model(adaptables = Resource.class)
public interface NewModel2 {

    @Inject
    public String getTitle();

    @Inject
    public String getDescription();

}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Design Goals

- Entirely annotation driven. "Pure" POJOs.
- Use standard annotations where possible.
- Pluggable
- OOTB, support resource properties (via ValueMap), SlingBindings, OSGi services, request attributes
- Adapt multiple objects - minimal required Resource and SlingHttpServletRequest
- Client doesn't know/care that these objects are different than any other adapter factory
- Support both classes and interfaces.
- Work with existing Sling infrastructure (i.e. not require changes to other bundles).

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Adobe

Timeline

- December 2013 – YAMF prototype announced on sling-dev
- January 2014 – API formalized and renamed to Sling Models
- Feburary 2014 – 1.0.0 release; Included in AEM 6.0 Beta
- March 2014 – 1.0.2 release; Included in AEM 6.0 Release
- May 2014 – 1.0.4 release; Memory leak bug fix.

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Adobe

Usage – What can be injected?

- In order…
  - SlingBindings objects
  - ValueMap properties (with Resource -> ValueMap adaptation)
  - Child Resources
  - Request Attributes
  - OSGi Services

- This is just the default set.

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - Annotations

- @org.apache.sling.models.annotations.Model
- @javax.inject.Inject
- @javax.inject.Named
- @org.apache.sling.models.annotations.Optional
- @org.apache.sling.models.annotations.Source
- @org.apache.sling.models.annotations.Filter
- @javax.inject.PostConstruct
- @org.apache.sling.models.annotations.Via
- @org.apache.sling.models.annotations.Default

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Adobe

Usage - @Model

- @Model(adaptables = Resource.class)
- @Model(adaptables = SlingHttpServletRequest.class)
- @Model(adaptables = { Resource.class, ValueMap.class })

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - @Inject

- @Inject private String title;
  - valueMap.get(“title”, String.class);
- @Inject public String getTitle();
  - valueMap.get(“title”, String.class);
- @Inject private String[] columnNames;
  - valueMap.get(“columnNames”, String[].class);
- @Inject private List<Filter> filters;
  - bundleContext.getServiceReferences(“javax.servlet.Filter”)

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Adobe

Usage - @Named

- By default, the name of the field or method is used to perform the injection.
- @Inject @Named(“jcr:title”) private String title;
  - valueMap.get(“jcr:title”, String.class);

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - @Optional

- By default, all @Inject points are required.
- resource.adaptTo(Model.class) <- returns null
- @Inject @Optional private String title;

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - @Source

- request.getAttribute(“text”) <- returns “goodbye”
- slingBindings.get(“text”) <- returns “hello”

- @Inject private String text; <- “hello” (SlingBindings is checked first)
- @Inject @Source(“request-attributes”) private String text; <- “goodbye”

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Adobe

Usage - @Source

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - @Filter

- Specifically for OSGi services:
- @Inject @Filter("(sling.servlet.extensions=json)") private List<Servlet> servlets;

- Implicitly applies @Source(“osgi-services”)

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - @PostConstruct

- @Inject private String text;
- @PostConstruct protected void doSomething() { log.info("text = {}", text); };

- Superclass @PostConstruct methods called first.

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - @Via

```
@Model(adaptables = SlingHttpServletRequest.class)
public class ViaModel {

    @Inject
    @Via("resource")
    private String firstProperty;

    public String getFirstProperty() {
        return firstProperty;
    }

}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage - @Default

- @Inject @Default(values="default text") private String text;
- Also
  - booleanValues
  - doubleValues
  - floatValues
  - intValues
  - longValues
  - shortValues

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage – Constructor Injection

- If you need the adaptable itself

```
@Model(adaptables = SlingHttpServletRequest.class)
public class WithOneConstructorModel {
    private final SlingHttpServletRequest request;
    @Inject
    private int attribute;

    public WithOneConstructorModel(SlingHttpServletRequest request) {
        this.request = request;
    }
    public int getAttribute() {
        return attribute;
    }
    public SlingHttpServletRequest getRequest() {
        return request;
    }
}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage – Child Adaptation

```
@Model(adaptables = Resource.class)
public interface ChildValueMapModel {

    @Inject
    public ValueMap getFirstChild();

}
```

- resource.getChild(“firstChild”).adaptTo(ValueMap.class)

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Usage – Fancy Child Adaptation

```
@Model(adaptables = Resource.class)
public interface ParentModel {

    @Inject
    public ChildModel getFirstChild();
}
```

- Works even if resource.adaptTo(ChildModel.class) isn't done by Sling Models

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Extensions – Custom Injectors

- Injectors are OSGi services implementing the org.apache.sling.models.spi.Injector interface
- Object getValue(Object adaptable, String name, Type type, AnnotatedElement element, DisposalCallbackRegistry callbackRegistry)
- adaptable – the object being adapted
- name – the name (either using @Named or the default name)
- element – the method or field
- callbackRegistry – Injector gets notified when the adapted model is garbage collected

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Adobe

Extensions – Custom Injector

```
public Object getValue(Object adaptable, String name,
        Type type, AnnotatedElement element,
        DisposalCallbackRegistry callbackRegistry) {
    Resource resource = getResource(adaptable);
    if (resource == null) {
        return null;
    } else if (type instanceof Class<?>) {
        InheritanceValueMap map = new
          HierarchyNodeInheritanceValueMap(resource);
        return map.getInherited(name, (Class<?>) type);
    } else {
        return null;
    }
}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Extensions – Custom Annotation

- Some injectors need extra data
  - Example: OSGi service filters

```
@Target({ ElementType.FIELD, ElementType.METHOD })
@Retention(RetentionPolicy.RUNTIME)
@Qualifier
@Source("resource-path")
public @interface ResourcePath {

    String value();

}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Extensions – Custom Annotations

```
public Object getValue(Object adaptable, String name,
        Type declaredType, AnnotatedElement element,
        DisposalCallbackRegistry callbackRegistry) {
    ResourcePath path =
        element.getAnnotation(ResourcePath.class);
    if (path == null) {
        return null;
    }
    ResourceResolver resolver = getResourceResolver(adaptable);
    if (resolver == null) {
        return null;
    }
    return resolver.getResource(path.value());
}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Extensions – Custom Annotations

```
@Model(adaptables = Resource.class)
public interface ResourcePathModel {

    @Inject @ResourcePath("/content/dam")
    Resource getResource();
}
```

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Availability

- Bundles can be downloaded from http://sling.apache.org/downloads.cgi
- Content Package can be downloaded from https://github.com/Adobe-Consulting-Services/com.adobe.acs.bundles.sling-models/releases
- Bleeding edge code can be built from http://svn.apache.org/repos/asf/sling/trunk/bundles/extensions/models

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.

Adobe

Future Roadmap

- Custom Annotations
- More Standard Injectors
  - Grandchild Resource Lists
- AEM-specific injectors in ACS AEM Commons
- Pluggable @Via support

© 2013 Adobe Systems Incorporated. All Rights Reserved. Adobe Confidential.


Adobe