Sling Models

In simple terms Sling Models are simple POJO classes which are mapped automatically with Sling Objects (resource, request objects..) and allow us to access jcr node property values directly into java classes.

* Entirely annotation driven(Need to write less code).
* Can adapt multiple objects –  – minimal required Resource and SlingHttpServletRequest  
    
  Support both classes and interfaces.
* Plugabble.
* Pure POJO classes.

org.apache.sling org.apache.sling.models.api   
  
Creating an adaptable class by annotation  
  
// Traditional way of adapting to resource using java class

Resource r = getResource();

return r.adaptTo(customClassName.class);

// Creating adaptable resource class using sling annotation

@Model(adaptables = Resource.class)

public class customClassName{ ... }  
  
Injecting properties in POJO class  
  
// Traditional way to getting values from jcr nodes in Java Class.

@Component

@Service

@Properties({

@Property(name = "adaptables", value = {"Resource" }),

@Property(name = "adapters", value = {"YourCustom" })

})  
  
// Retrieving vales from jcr node through inject in Sling Model

@Model(adaptables = Resource.class)

public class slingModelExample {

@Inject // email property is always looked from Resource( after adapting to ValueMap), if this property value

is not present then it returns null. If this property is in itself not available then it throws exception.

private String email;

@Inject @Optional // It is not mandatory to have firstname property in Resource.

private String firstName;

// Use @Named if jcr node property name is different than class variable.

use @Default if you want to assign default values, If empty then assign default value as 'NA'

@Inject @Named("surname") @Default(values="NA")

private String lastName;

}

@Inject // OSGi Service

private Externalizer externalizer;

//@PostConstructor annotation defines that this method will run after injecting all field

@PostConstruct

protected void init() {

// gets executed after the class is created and all variables are injected

}

So, just to summarize above code:-

@Inject :- email property is always looked from Resource( after adapting to ValueMap), if this property value is not present then it returns null. If this property is in itself not available then it throws exception.

@Inject @Optional :- It is not mandatory to have firstname property in Resource.

@Named :- Use  it if jcr node property name is different than class variable.

@Default :- use it if you want to assign default values, If empty then assign default value as ‘NA’.

@PostConstructor:-  annotation defines that this method will run after injecting all field.

<dependency>

<groupId>org.apache.sling</groupId>

<artifactId>org.apache.sling.models.api</artifactId>

</dependency>