

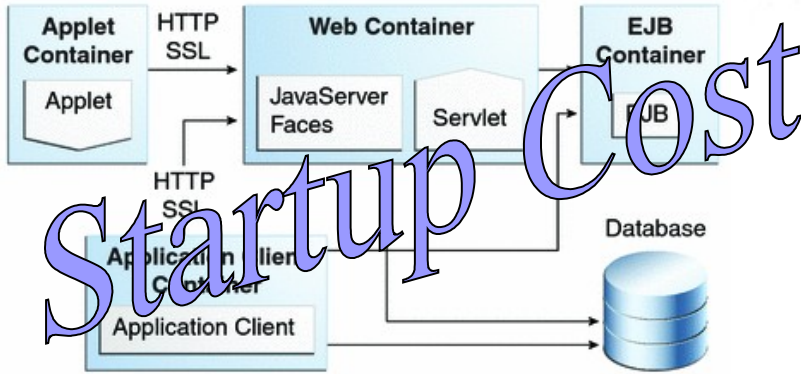


Forging a Bond between Eclipse and the Command-line

Agenda

- A bit of Forge history
- Command-line vs IDE
- Demo: Forge in Eclipse
- Under the hood
- Future directions

Daunting? What Do You Mean?



Startup Cost

Details

Application Client Container	Java Persistence	Java SE
	Management	
	WS Metadata	
	Web Services	
	Application	
	JAX-WS	SAAJ
	JAX-RPC	

Web Container	JSR 330	Java SE
	Interceptors	
	Managed Beans	
	JSR 299	
	Bean Validation	
	EJB Lite	
	EL	
	JavaMail	
	JSP	
	JavaServer Faces	
	Connectors	Java SE
	Java Persistence	
	JMS	
	Management	
	WS Metadata	
	Web Services	
	JACC	
	JAX-WS	
	JAX-RPC	
	JAX-WS	
	JAX-RPC	

Integration

EJB Container	JSR 330	Java SE
	Interceptors	
	Managed Beans	
	JSR 299	
	Bean Validation	
	EJB	
	Java Persistence	
	JTA	
	Connectors	
	JMS	
	Management	Java SE
	WS Metadata	
	Web Services	
	JACC	
	JASPIC	
	JAXR	
	JAX-RS	
	JAX-WS	
	JAX-RPC	
	JAX-WS	
	JAX-RPC	

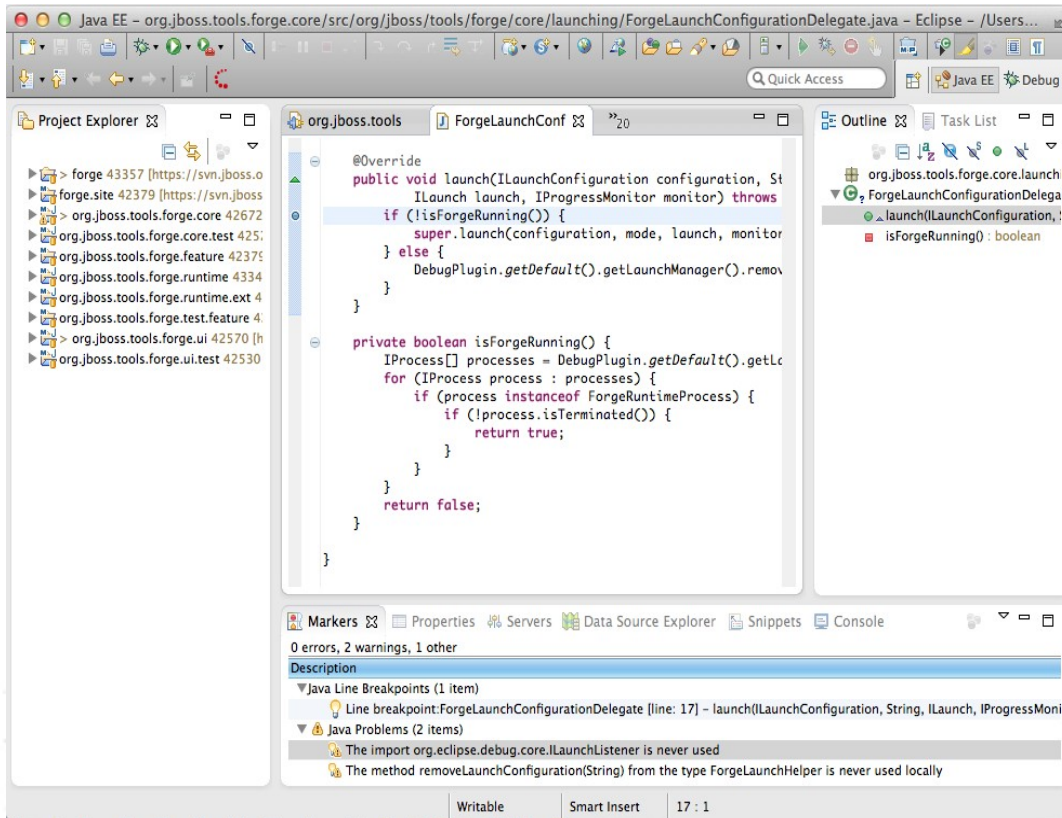
Gotchas

<http://jboss.org/forged>



Carve out a project, work it into shape

But, I Want My ~~MTV~~ IDE



- Incremental builds!
- Workspace navigation!
- Outline view!
- Debugging!
- Form based editing!

Have Your Cake and Eat It!

The screenshot displays the Eclipse IDE interface. The main editor shows the `Customer.java` file with the following code:

```
package com.example.foo.model;

import javax.persistence.Entity;
import java.io.Serializable;
import javax.persistence.Id;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Column;
import javax.persistence.Version;
import java.lang.Override;

@Entity
public class Customer implements Serializable
{
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id", updatable = false, nullable = false)
    Long id = null;

    @Version
    private @Column(name = "version")
    int version = 0;

    @Column
    private String firstName;

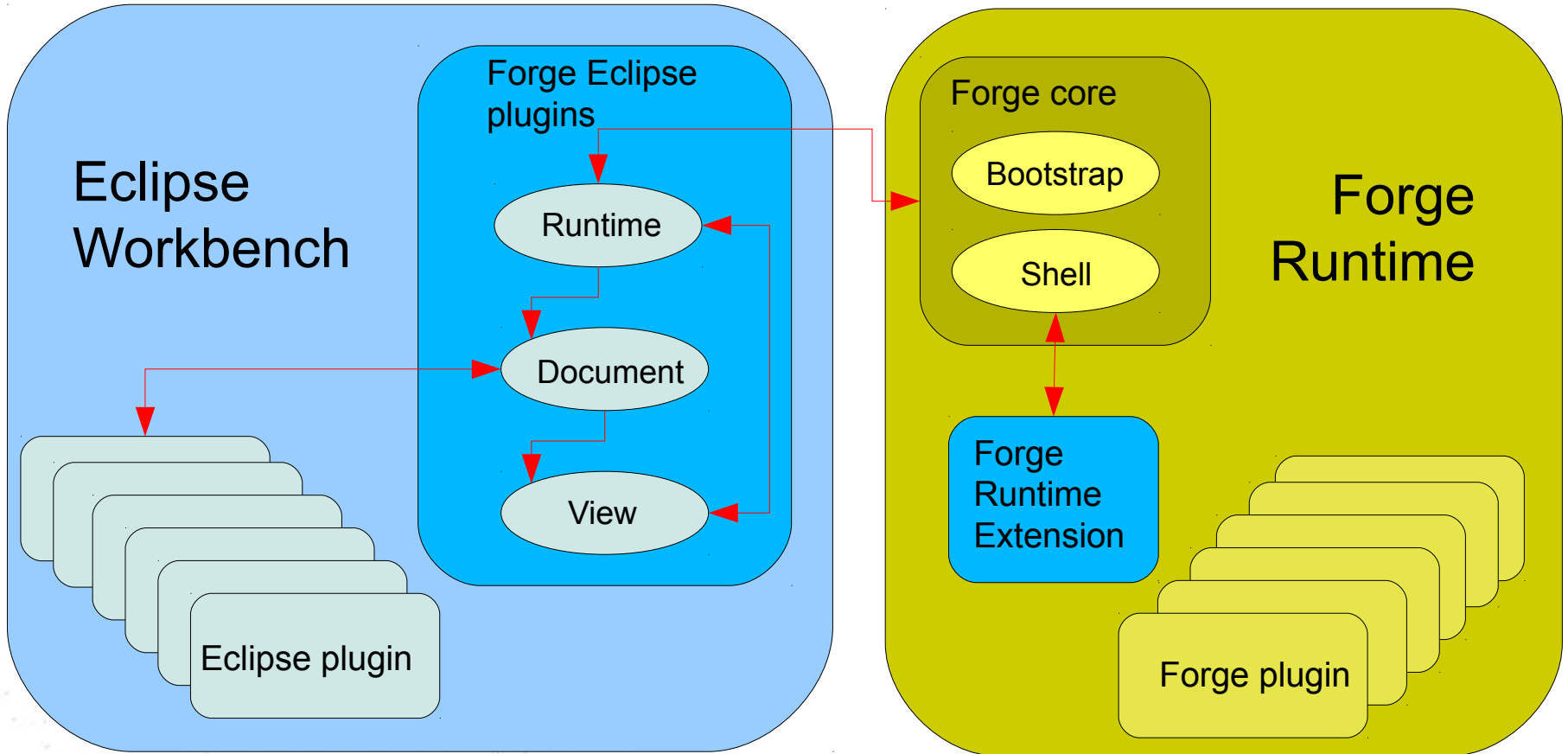
    public Long getId()
    {
        ...
    }
}
```

The Project Explorer on the left shows the project structure, including `persistence.xml`. The Outline on the right shows the class structure with fields like `id : Long`, `version : int`, and `firstName : String`. The Forge Console at the bottom shows the output of the `fi` command:

```
Wrote /Users/koen/delft/workspace/foo/src/main/java/com/example/foo/model/Customer.java
[foo] Customer.java $ fi
fingerprint find field
[foo] Customer.java $ field
oneToMany manyToMany oneToOne int string temporal number boolean long
manyToOne custom
[foo] Customer.java $ field string --named firstName
Added field to com.example.foo.model.Customer: @Column private String firstName;

Wrote /Users/koen/delft/workspace/foo/src/main/java/com/example/foo/model/Customer.java
[foo] Customer.java $
```

How Was the Cake Baked?



Future Cake Recipes

- Easier extension for Forge plugin writers
 - Declarative command UI
- No more multiplexing on character input/output
 - JMS?
- Tighter integration with the Eclipse workbench
 - OSGi?

Conclusion

- Old school Forge
 - Gets you started **quickly**
 - Takes care of technological **gotchas**
 - Adds/activates technologies by means of **plugins**
- Eclipse Forge
 - Integrates this power into the **Eclipse workbench**
 - Provides the **best of both worlds**

Questions?



<http://jboss.org/forged>