What's new in WildFly 8 ?

Arun Gupta · Red Hat · @arungupta

Arun Gupta

- Director, Developer Advocacy, Red Hat Inc.
- O'Reilly and McGraw Hill author
- Fitness freak

The Great Java Application Server Debate with Tomcat, JBoss, GlassFish, Jetty and Liberty Profile

May 21, 2013 | Simon Maple | 27 comments

Tweet 171 | Like 196 | +1 105 | Share 60 | DZone 11 0

Part V – ...And The Best Application Server Award Goes To...

In case you were wondering, we did finally decide that one application server among those tested proved to win over the others...JBoss wins the award!

JBoss AS 7 (aka WildFly) **WINS THE AWARD!**

> *If we had to pick a **winner**, it would be **JBoss**. The only application server in the group whose score **never dropped below a 4***
>
> — zeroturnaround.com

> *JBoss **consistently** performs **very well** in each category which is why it also **shines** in the developer profiles exercise*
>
> — zeroturnaround.com


WildFly
The new and improved JBoss Application Server!

`Objective:`

Understand the new features of WildFly 8 and revise some of the features carry forward from AS 7.x.

What is WildFly 8 ?

What is WildFly 8 ?

- Previously called “JBoss Application Server”

What is WildFly 8 ?

- Previously called “JBoss Application Server”
- **Upstream for Red Hat JBoss Enterprise Application Platform (JBoss EAP)**

What is WildFly 8 ?

- Previously called “JBoss Application Server”
- Upstream for Red Hat JBoss Enterprise Application Platform (JBoss EAP)
- **Fast, lightweight, manageable**

What is WildFly 8 ?

- Previously called “JBoss Application Server”
- Upstream for Red Hat JBoss Enterprise Application Platform (JBoss EAP)
- Fast, lightweight, manageable
- Developer friendly

What is WildFly 8 ?

- Previously called “JBoss Application Server”
- Upstream for Red Hat JBoss Enterprise Application Platform (JBoss EAP)
- Fast, lightweight, manageable
- Developer friendly
- Supports Java EE standards and beyond

What is WildFly 8 ?

- Previously called “JBoss Application Server”
- Upstream for Red Hat JBoss Enterprise Application Platform (JBoss EAP)
- Fast, lightweight, manageable
- Developer friendly
- Supports Java EE standards and beyond
- Open source

WildFly 8 main features

WildFly 8 main features

- Java EE7 support

WildFly 8 main features

- Java EE7 support
- High performance web server Undertow

WildFly 8 main features

- Java EE7 support
- High performance web server Undertow
- **Reduced port usage**

WildFly 8 main features

- Java EE7 support
- High performance web server Undertow
- Reduced port usage
- Role based administration control & auditing

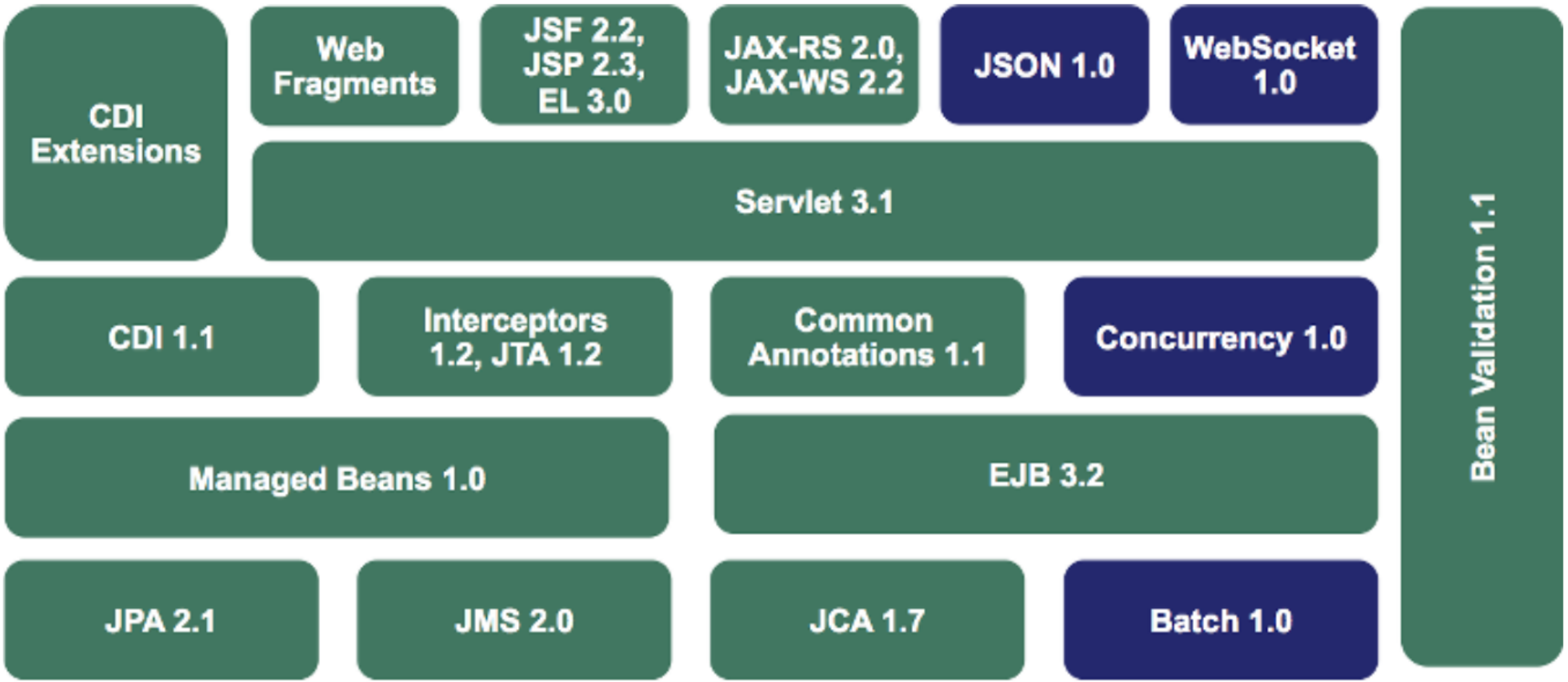
WildFly 8 main features

- Java EE7 support
- High performance web server Undertow
- Reduced port usage
- Role based administration control & auditing
- **Automated patching**

WildFly 8 main features

- Java EE7 support
- High performance web server Undertow
- Reduced port usage
- Role based administration control & auditing
- Automated patching
- Minimalistic "core" distribution

WildFly: Java EE 7



WildFly: Java EE 7 WebSocket

ChatServer.java

```
@ServerEndpoint("/chat") (1)
public class ChatEndpoint {
    @OnMessage (2)
    public void message(String message,
                        Session client) (3)
                    throws IOException, EncodeException {
        for (Session peer : client.getOpenSessions()) {
            peer.getBasicRemote().sendText(message);
        }
    }
}
```

1. Creates a WebSocket endpoint, defines the listening URL
2. Marks the method that receives incoming WebSocket message
3. Payload of the WebSocket message

`job.xml`

```
<job id="myJob" xmlns="http://xmlns.jcp.org/xml/ns/javaee" version="1.0">
    <step id="myStep" >
        <chunk item-count="3"> (1)
            <reader ref="myItemReader"/> (2)
            <processor ref="myItemProcessor"/> (3)
            <writer ref="myItemWriter"/> (4)
        </chunk>
    </step>
</job>
```

1. Item-oriented processing, number of items in chunk
2. Item reader for chunk processing
3. Item processor for chunk processing
4. Item writer for chunk processing

CreateJson.java

```
JsonObject jsonObject = Json.createObjectBuilder() (1)
                .add("apple", "red") (2)
                .add("banana", "yellow")
                .build(); (3)
StringWriter w = new StringWriter();
JsonWriter writer = Json.createWriter(w); (4)
writer.write(jsonObject);
```

1. Creates a JSON object builder
2. Adds a name/value pair to the JSON object
3. Returns the JSON object associated with this builder
4. Writes the JSON object to the writer

WildFly: Java EE 7 Concurrency

`RunMyTask.java`

```
public class MyTask implements Runnable { (1)

    @Override
    public void run() {
        . . .
    }
}

@Resource(name = "DefaultManagedExecutorService") (2)
ManagedExecutorService defaultExecutor;

executor.submit(new MyTask()); (3)
```

(1) `Runnable` or `Callable` tasks can be submitted

(2) `ManagedExecutor` is injected, default resource provided

(3) Submit the task

`RunClient.java`

```
Client client = ClientBuilder.newClient(); (1)
WebTarget target = client.target("..."); (2)
target.register(Person.class);
Person p = target
              .path("{id}") (3)
              .resolveTemplate("id", "1")
              .request(MediaType.APPLICATION_XML) (4)
              .get(Person.class); (5)
```

1. `ClientBuilder` is the entry point
2. Build a new web resource target, specifies the path
3. Sub resource URI
4. Define the accepted response media types
5. Call HTTP GET, specify the type of resource

WildFly: Java EE 7 JMS

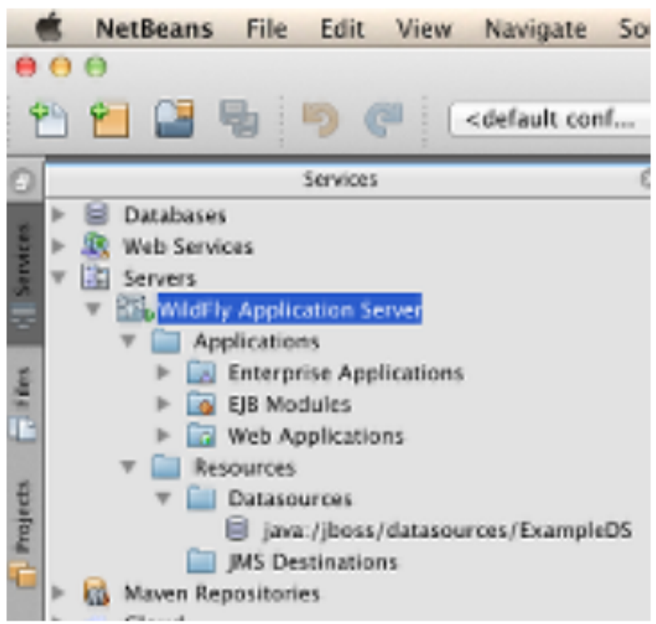
SendMessage.java

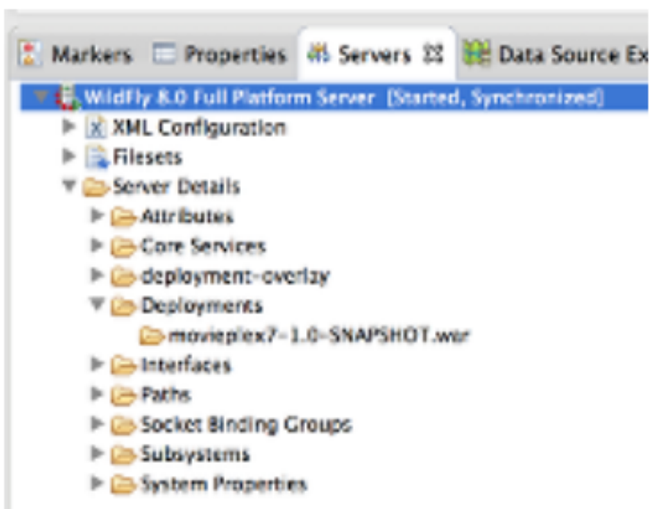
```
@JMSDestinationDefinition(name="myQueue", interfaceName="javax.jms.Queue") (1)

@Resource(mappedName="myQueue")
Queue syncQueue;

@Inject
// @JMSConnectionFactory("java:comp/DefaultJMSConnectionFactory") (2)
private JMSContext context; (3)
```

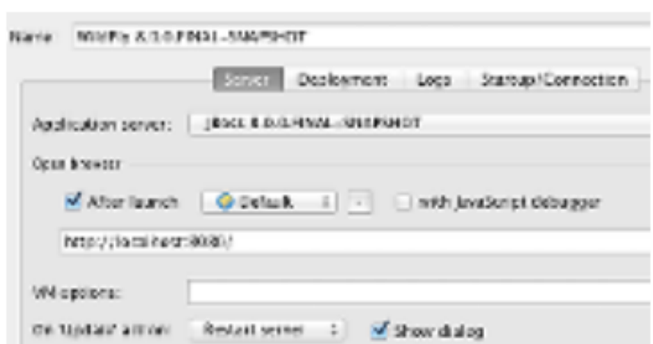
```
context.createProducer().send(syncQueue, "..."); (4)
```

1. Create destination resource during deployment
2. Default JMS connection factory
3. Main interface of the simplified API
4. Fluent builder API, runtime exceptions

WildFly: Java EE 7 IDEs







WildFly: New web server (Undertow)

WildFly: New web server (Undertow)

- **Flexible and high-performance**

WildFly: New web server (Undertow)

- Flexible and high-performance
- Blocking / non-blocking based on NIO

WildFly: New web server (Undertow)

- Flexible and high-performance
- Blocking / non-blocking based on NIO
- Composition/handler based architecture

WildFly: New web server (Undertow)

- Flexible and high-performance
- Blocking / non-blocking based on NIO
- Composition/handler based architecture
- **Lightweight & fully embeddable**

WildFly: New web server (Undertow)

- Flexible and high-performance
- Blocking / non-blocking based on NIO
- Composition/handler based architecture
- Lightweight & fully embeddable
- **Supports Servlet 3.1 & HTTP upgrade**

WildFly: New web server (Undertow)

- Flexible and high-performance
- Blocking / non-blocking based on NIO
- Composition/handler based architecture
- Lightweight & fully embeddable
- Supports Servlet 3.1 & HTTP upgrade
- `mod_cluster` supported

WildFly new web server: Undertow

NonBlockingHandler.java

```
Undertow.builder() (1)
   .addListener(8080, "localhost")
   .setHandler(new HttpHandler() { (2)
     @Override
     public void handleRequest(final HttpServerExchange exchange)
       throws Exception {
       exchange.getResponseHeaders()
               .put(Headers.CONTENT_TYPE, "text/plain");
       exchange.getResponseSender()
               .send("Hello World");
     }
   }).build().start(); (3)
```

(1) Same API used for WildFly integration, fluent builder API

(2) Can create multiple handlers

(3) Start the handler in JVM

Undertow benchmarks

```
techempower@lg01:~$ wrk -d 30 -c 256 -t 40 http://10.0.3.2:8080/byte
Running 30s test @ http://10.0.3.2:8080/byte
  40 threads and 256 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   247.05us    3.52ms 624.37ms   99.90%
    Req/Sec    27.89k     6.24k   50.22k    71.15%
  31173283 requests in 29.99s, 3.83GB read
  Socket errors: connect 0, read 0, write 0, timeout 9
Requests/sec: 1039305.27
Transfer/sec:    130.83MB
```

This is output from Wrk testing a single server running Undertow using conditions similar to Google's test (1-byte response body, no HTTP pipelining, no special request headers) **1.039 million requests per second**.

- http://www.techempower.com/blog/2014/03/04/one-million-http-rps-without-load-balancing-is-easy/

WildFly: Port reduction

WildFly: Port reduction

- Uses HTTP Upgrade

WildFly: Port reduction

- Uses HTTP Upgrade
- Number of ports in default installation is two
  - 8080 for applications
  - 9990 for management

WildFly: Port reduction

- Uses HTTP Upgrade
- Number of ports in default installation is two
  - 8080 for applications
  - 9990 for management
- **Only overhead is the initial HTTP Upgrade request/response**

WildFly: Role based access control

WildFly: Role based access control

- **Pre-defined administrative and privileged Roles**
  - **Monitor, Operator, Maintainer, Deployer, Administrator, Auditor, Super User**

WildFly: Role based access control

- Pre-defined administrative and privileged Roles
  - Monitor, Operator, Maintainer, Deployer, Administrator, Auditor, Super User
- Roles is a set of Permissions

WildFly: Role based access control

- Pre-defined administrative and privileged Roles
  - Monitor, Operator, Maintainer, Deployer, Administrator, Auditor, Super User
- Roles is a set of Permissions
- Permissions specify which Actions (lookup, read, write) are allowed on resources

WildFly: Role based access control

- Pre-defined administrative and privileged Roles
  - Monitor, Operator, Maintainer, Deployer, Administrator, Auditor, Super User
- Roles is a set of Permissions
- Permissions specify which Actions (lookup, read, write) are allowed on resources
- Users or Groups are defined in Roles

WildFly: Administrative audit logging

WildFly: Administrative audit logging

- Logging of connection/authentication events

WildFly: Administrative audit logging

- Logging of connection/authentication events
- **Logging of management operations**

WildFly: Administrative audit logging

- Logging of connection/authentication events
- Logging of management operations
- **Log message as JSON records**

WildFly: Administrative audit logging

- Logging of connection/authentication events
- Logging of management operations
- Log message as JSON records
- **Audit logging handlers**
  - **Local file**
  - **Syslog (UDP / TCP / TLS)**

WildFly: Automated patching

- Allows libraries and configuration updates in an installation

WildFly: Automated patching

- Allows libraries and configuration updates in an installation
- **Patches are zip bundles with updates and metadata**

WildFly: Automated patching

- Allows libraries and configuration updates in an installation
- Patches are zip bundles with updates and metadata
- **Multiple one-off patches can be applied; invalidated by the next point/CP release**

WildFly: Automated patching

- Allows libraries and configuration updates in an installation
- Patches are zip bundles with updates and metadata
- Multiple one-off patches can be applied; invalidated by the next point/CP release
- Rollbacks are possible

WildFly: Minimalistic "core" distribution

WildFly: Minimalistic "core" distribution

- 15 MB download

WildFly: Minimalistic "core" distribution

- 15 MB download
- Rich management layer

WildFly: Minimalistic "core" distribution

- 15 MB download
- Rich management layer
- **Fully concurrent service container with advanced capabilities**

WildFly: Minimalistic "core" distribution

- 15 MB download
- Rich management layer
- Fully concurrent service container with advanced capabilities
- **Modular class loading which enables multi-tenancy of applications**

WildFly: Minimalistic "core" distribution

- 15 MB download
- Rich management layer
- Fully concurrent service container with advanced capabilities
- Modular class loading which enables multi-tenancy of applications
- Pluggable hot deployment layer

WildFly: Minimalistic "core" distribution

- 15 MB download
- Rich management layer
- Fully concurrent service container with advanced capabilities
- Modular class loading which enables multi-tenancy of applications
- Pluggable hot deployment layer
- **Built-in lightweight web server**

WildFly: Miscellaneous

- Improved JDK8 compatibility

WildFly: Miscellaneous

- Improved JDK8 compatibility
- RESTEasy 3

WildFly: Miscellaneous

- Improved JDK8 compatibility
- RESTEasy 3
- Hibernate search

WildFly: Miscellaneous

- Improved JDK8 compatibility
- RESTEasy 3
- Hibernate search
- **Per-deployment security permissions**

WildFly: Miscellaneous

- Improved JDK8 compatibility
- RESTEasy 3
- Hibernate search
- Per-deployment security permissions
- New public clustering API

WildFly: Miscellaneous

- Improved JDK8 compatibility
- RESTEasy 3
- Hibernate search
- Per-deployment security permissions
- New public clustering API
- Pruned: CMP, JAX-RPC, JSR 88

WildFly: In the cloud

Carry forward from AS 7.x

- Standalone and Managed Domain
- Centralized Administration
  - Command Line Interface (jboss-cli)
  - Admin Console
  - Configuration files

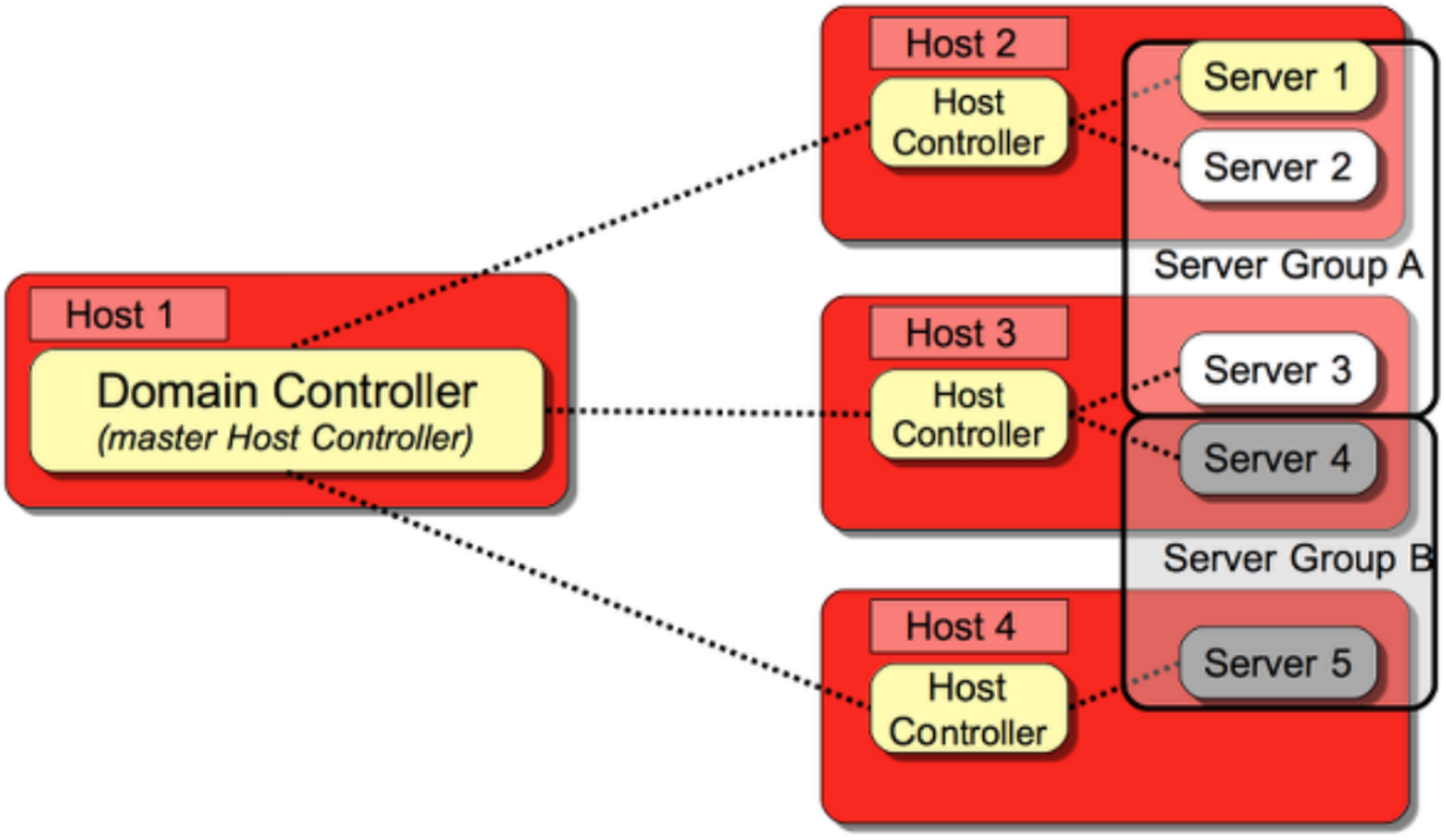
Standalone and Managed Domain

- **Standalone**: single independent instance

Standalone and Managed Domain

- Standalone: single independent instance
- Managed domain: manage multiple WildFly instances from a single control point
  - Host controller
  - Domain controller
  - Server group
  - Server

Managed Domain



Command Line Interface

Command Line Interface

- `jboss-cli.sh|bat`

Command Line Interface

- `jboss-cli.sh|bat`
- **Connects to standalone instance or Domain controller**

Command Line Interface

- `jboss-cli.sh|bat`
- Connects to standalone instance or Domain controller
- Interactive mode: *nix-style shell
  - Contextual command and resource-tab completion

Command Line Interface

- `jboss-cli.sh|bat`
- Connects to standalone instance or Domain controller
- Interactive mode: *nix-style shell
  - Contextual command and resource-tab completion
- **Non-interactive mode: commands in files**

Command Line Interface

- `jboss-cli.sh|bat`
- Connects to standalone instance or Domain controller
- Interactive mode: *nix-style shell
  - Contextual command and resource-tab completion
- Non-interactive mode: commands in files
- High-level compound operations

Command Line Interface

- `jboss-cli.sh|bat`
- Connects to standalone instance or Domain controller
- Interactive mode: *nix-style shell
  - Contextual command and resource-tab completion
- Non-interactive mode: commands in files
- High-level compound operations
- Persistent changes

Admin Console

- Simple

Admin Console

- Simple
- Fast

Admin Console

- Simple
- Fast
- Lightweight

- Simple
- Fast
- Lightweight
- **Avoids XML configuration**

- Simple
- Fast
- Lightweight
- Avoids XML configuration
- **Single instance and domains**

Admin Console

- Simple
- Fast
- Lightweight
- Avoids XML configuration
- Single instance and domains
- **Mostly configuration, basic monitoring**
  - Not a Red Hat JBoss Operations Network replacement

- Java EE 7 compliant
- lightweight
- managable
- highly scalable
- open source
- application server

Now available!

Open source polyglot conference

References

- WildFly - http://wildfly.org, http://github.com/wildfly, @WildFlyAS
- JBoss EAP 6.2 - http://redhat.com/jboss
- Java EE 7 samples - https://github.com/javaee-samples/javaee7-samples
- Slides generated with Asciidoctor and DZSlides backend
- Original slide template - Dan Allen & Sarah White

Arun Gupta

@arungupta