

# What's new in WildFly 8 ?

---

Arun Gupta · Red Hat · [@arungupta](https://twitter.com/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



## 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](http://zeroturnaround.com)

“JBoss ***consistently*** performs ***very well*** in each category which is why it also ***shines*** in the developer profiles exercise

— zeroturnaround.com

# WildFly



JBoss Application Server  
has a new name...  
and it's even  
**@#\$%ing**  
faster!

[Learn more about WildFly >](#)

## **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

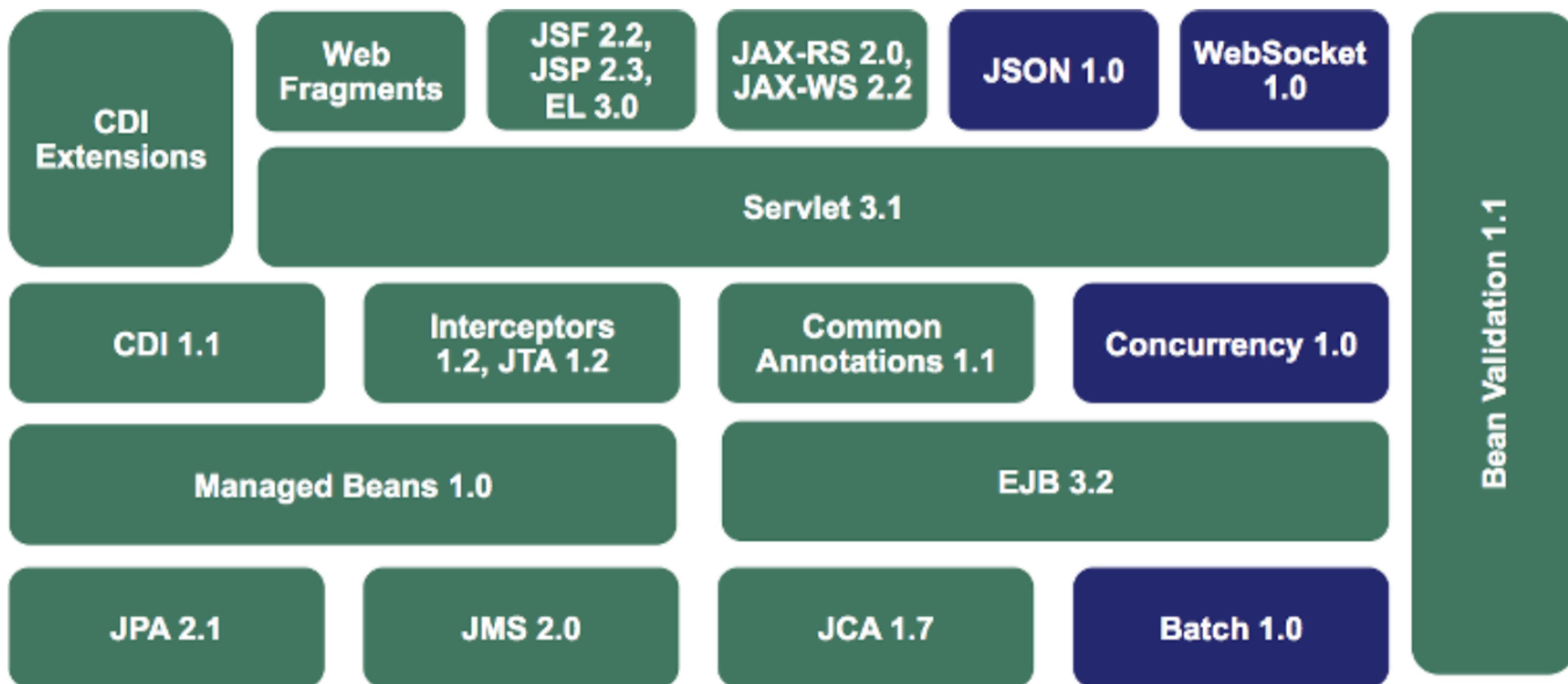
## WildFly 8 main features

- 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: 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

# WildFly: Java EE 7 Batch

## 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

# WildFly: Java EE 7 JSON

## 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

# WildFly: Java EE 7 JAX-RS

## 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/DefaultJMSSConnectionFactory") ②  
private JMSContext context; ③  
  
context.createProducer().send(syncQueue, "..."); ④
```

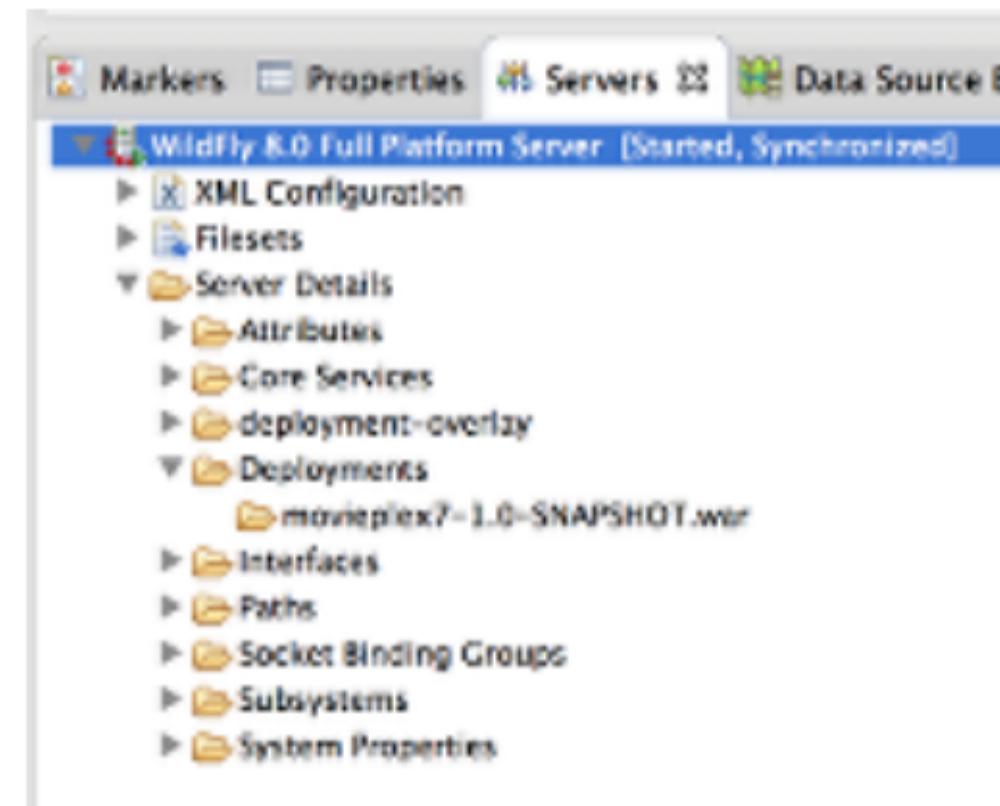
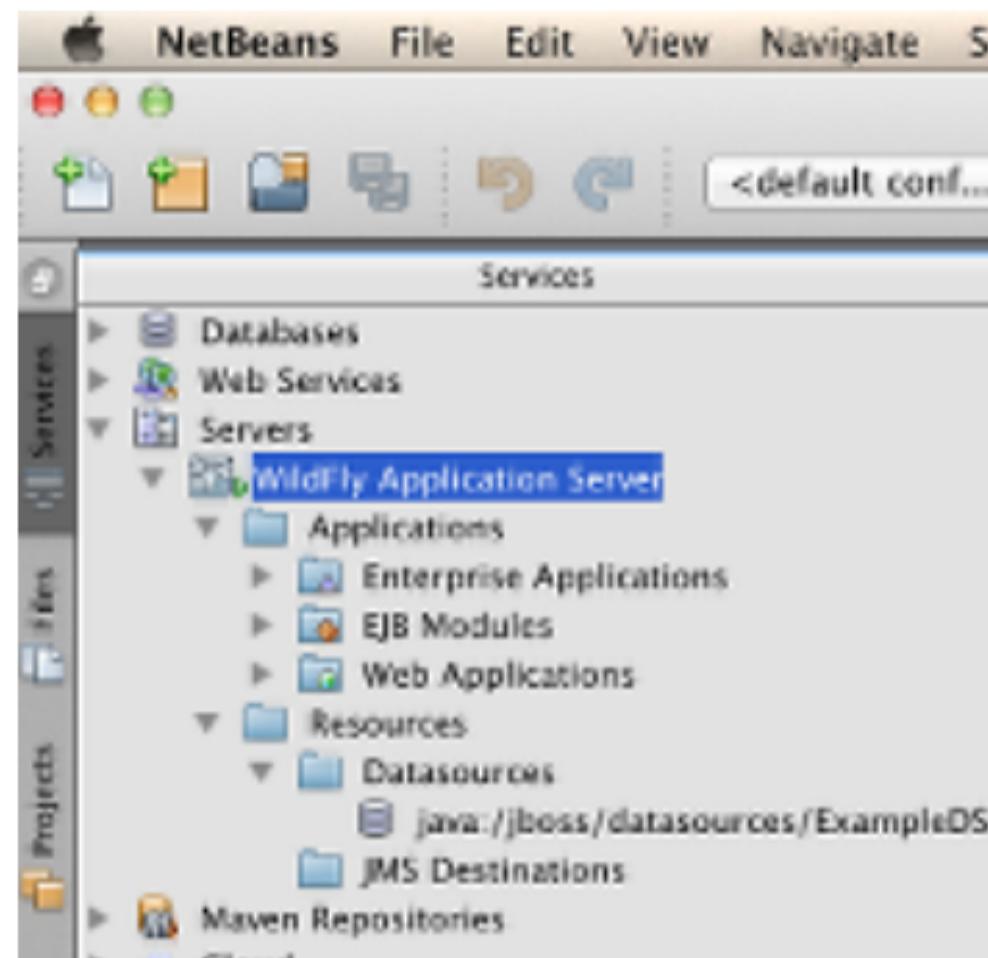
① Create destination resource during deployment

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 NIO based APIs**



## **WildFly: New web server (**Undertow**)**

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



## **WildFly: New web server (**Undertow**)**

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



## **WildFly: New web server (**Undertow**)**

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

# 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

# WildFly: Port reduction

---

## WildFly: Port reduction

- HTTP Upgrade to reduce the number of ports in the default installation to just two
  - 8080 for applications with JNDI and EJB multiplexed
  - 9990 for management, for both HTTP/JSON & Native API

## WildFly: Port reduction

- HTTP Upgrade to reduce the number of ports in the default installation to just two
  - 8080 for applications with JNDI and EJB multiplexed
  - 9990 for management, for both HTTP/JSON & Native API
- 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**

---

## 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: 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 and Managed Domain

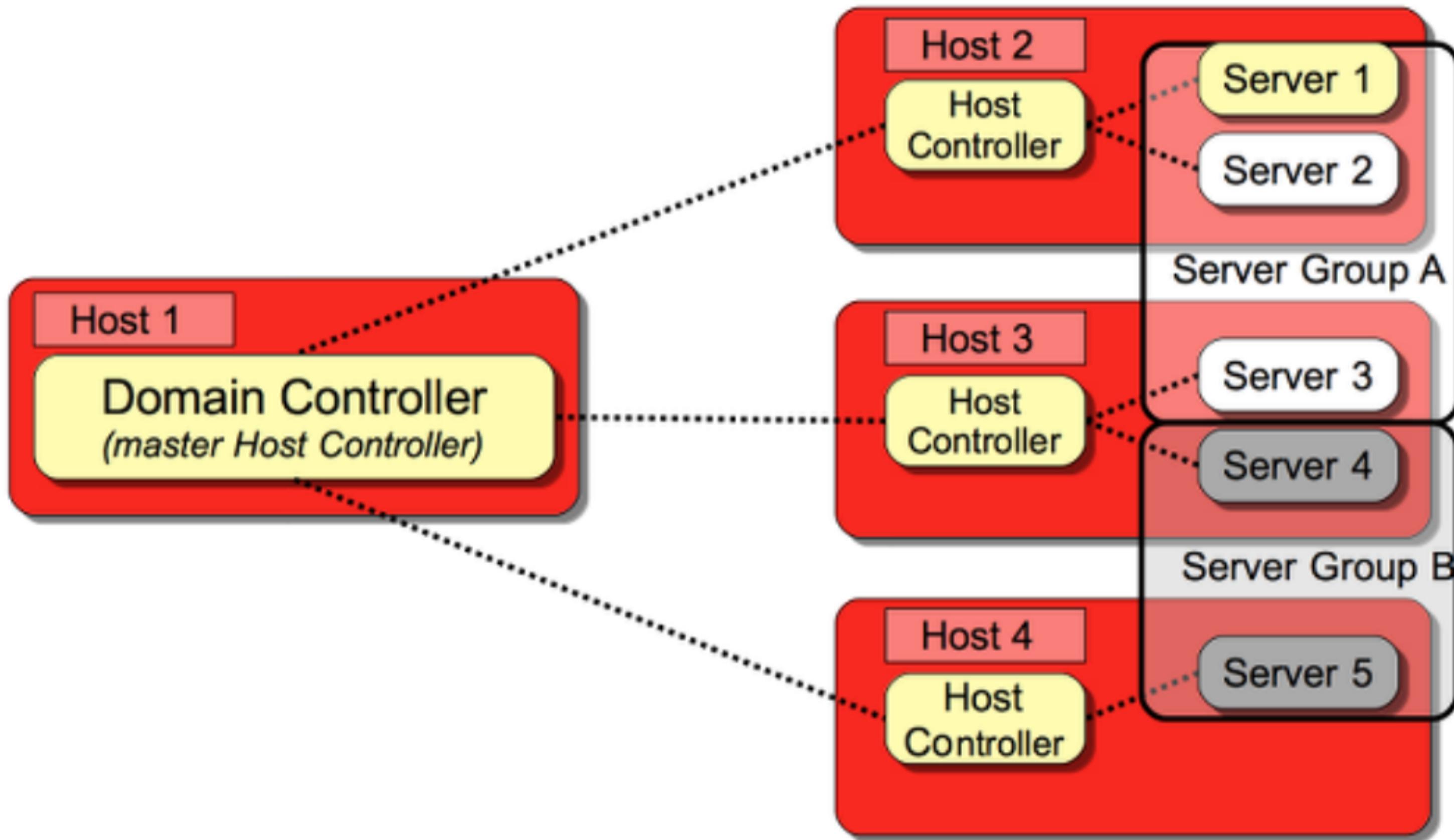
- **Standalone:** single independent instance

## **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
- \*nix-style shell syntax, resources as files

## Command Line Interface

- `jboss-cli.sh|bat`
- **Connects to standalone instance or Domain controller**
- **\*nix-style shell syntax, resources as files**
- **Contextual command and resource-tab completion**

## Command Line Interface

- `jboss-cli.sh|bat`
- **Connects to standalone instance or Domain controller**
- **\*nix-style shell syntax, resources as files**
- **Contextual command and resource-tab completion**
- **High-level compound operations**

## Command Line Interface

- `jboss-cli.sh|bat`
- **Connects to standalone instance or Domain controller**
- **\*nix-style shell syntax, resources as files**
- **Contextual command and resource-tab completion**
- **High-level compound operations**
- **Persistent changes**

# **Admin Console**

---

- Simple

## Admin Console

- Simple
- Fast



## **Admin Console**

- **Simple**
  - **Fast**
  - **Lightweight**
-

## Admin Console

- 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
- manageable
- highly scalable
- open source
- application server

**Now available!**



## 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