

# Real-time collaborative *editor* for AsciiDoc

When WebSocket met Asciidoctor

Maxime GREAU

# About Me

April 13-17, 2014 San Francisco, California

**DEVNATION**

# About Me

**IT Architect** Ministry for the Economy  
and Finance

# About Me

**IT Architect** Ministry for the Economy  
and Finance

**Author** Apache Maven (French Book)

# About Me

**IT Architect** Ministry for the Economy  
and Finance

**Author** Apache Maven (French Book)

**Technical Reviewer** WildFly - EE7

# About Me

**IT Architect** Ministry for the Economy  
and Finance

**Author** Apache Maven (French Book)

**Technical Reviewer** WildFly - EE7

**Blog** <http://mgreau.com>

# About Me

**IT Architect** Ministry for the Economy  
and Finance

**Author** Apache Maven (French Book)

**Technical Reviewer** WildFly - EE7

**Blog** <http://mgreau.com>

**Twitter** @mgreau **#adeditor**

# When WebSocket met Asciidoctor

```
 .o.          o8o   o8o   ooooooooooooo. 
 .888.        ``"  ``"  `888'   `Y8b 
 .8"888.      .oooo.o  .ooooo.  oooo  oooo  888   888   .ooooo.   .ooooo. 
 .8' `888.    d88(   "8 d88'  ``Y8  `888  `888  888   888  d88'  `88b d88'  ``Y8 
 .88oooo8888.  `Y88b.  888   888   888   888   888 888   888 888   888 
 .8'   `888.  o. )88b 888   .o8  888   888   888   d88' 888   888 888   .o8 
 o88o   o8888o 8""888P' `Y8bod8P' o888o o888o o888boood8P'  `Y8bod8P' `Y8bod8P'
```



**HTML**



**JS**



**WildFly**



**OPENSIFT**  
by Red Hat®



QUICK POLL

**Who sent an email today?**



**Who wrote technical  
documentation in the past two  
days ?**

“Writing documentation  
has to be **as easy as**  
writing email !”

Your solution is  
**Asciidoc**

**AsciiDoc is...**

**lightweight** **markup**  
**publishing** **toolchain**



# AsciiDoc : Lightweight markup

## = Document Title2

Doc Writer <doc@asciidoc.org>

v1.0, 2013-01-01: Initial version

<http://asciidoc.org>[AsciiDoc] is a lightweight markup language.

This is the optional preamble (an untitled section body), useful for writing simple sectionless documents consisting only of a preamble.

**NOTE:** The abstract, preface, appendix, bibliography, glossary and index section titles are significant (*\_specialsections\_*).

## == First section

Document sections start at \*level 1\* and can nest four levels deep.

- \* Item 1
- \* Item 2

**“**Use *AsciiDoc* for document markup. It’s actually *readable* by humans, *easier to parse* and more *flexible than XML*.

— Linus Torvald

# What's Asciidoctor?

# Asciidoctor

A modern, open source implementation of  
AsciiDoc in Ruby

# Output formats (i.e., backends)

# Output formats (i.e., backends)

HTML 5

# **Output formats (i.e., backends)**

**HTML 5**  
**DocBook 4.5 & 5.0**

# Output formats (i.e., backends)

HTML 5

DocBook 4.5 & 5.0

PDF fop, dblatex

# **Output formats (i.e., backends)**

**HTML 5**

**DocBook 4.5 & 5.0**

**PDF** fop, dblatex

**eBook** ePub 2, mobi

# Output formats (i.e., backends)

HTML 5

DocBook 4.5 & 5.0

PDF fop, dblatex

eBook ePub 2, mobi

slides deck.js, dzslides, reveal.js

# **Output formats (i.e., backends)**

**HTML 5**

**DocBook 4.5 & 5.0**

**PDF** fop, dblatex

**eBook** ePub 2, mobi

**slides** deck.js, dzslides, reveal.js

**man pages**

# **Output formats (i.e., backends)**

**HTML 5**

**DocBook 4.5 & 5.0**

**PDF** fop, dblatex

**eBook** ePub 2, mobi

**slides** deck.js, dzslides, reveal.js

**man pages**

**custom**

# How do I use Asciidoctor ?

# How do I use Asciidoctor ?

**Ruby** asciidoctor asciidoctor-pdf

# How do I use Asciidoctor ?

**Ruby** asciidoctor asciidoctor-pdf

**Java** asciidoctorJ, asciidoctor-maven-plugin

# How do I use Asciidoctor ?

**Ruby** asciidoctor asciidoctor-pdf

**Java** asciidoctorJ, asciidoctor-maven-plugin

**Groovy** asciidoctor-gradle-plugin

# How do I use Asciidoctor ?

**Ruby** asciidoctor asciidoctor-pdf

**Java** asciidoctorJ, asciidoctor-maven-plugin

**Groovy** asciidoctor-gradle-plugin

**Javascript** asciidoctor.js,

Chrome/Firefox extension

# Who is using Asciidoctor ?

**Frameworks** Spring, Infinispan  
**JSR** CDI Specifications  
**Publishers** O'Reilly  
**Repositories** Github, Bintray  
and more...

# Asciidoctor via asciidoctor.js

DEMO TIME

The screenshot shows a web browser window with multiple tabs at the top. The active tab is titled "Realtime Collaborative Edit" and has the URL "wildfly-mgreau.rhcloud.com/ad-editor/". A yellow banner at the top says "You work on OFFLINE MODE!". Below it, there are buttons for "Alt+R" (radio button), "Change" (radio button), "Diff", "Patch", "Save", and "Load". On the right, there are icons for "minimize", "maximize", and "close", along with a status bar showing "1472 ms" and "max".

**Left Panel (Raw AsciiDoc):**

```
5 :nodoc:
6 :idseparator: -
7 :online-demo: http://wildfly-mgreau.rhcloud.com/ad-editor/
8 :milestones: https://github.com/mgrehu/when-websocket-net-asciidoctor/issues/milestones
9 :issues: https://github.com/mgrehu/when-websocket-net-asciidoctor/issues
10 :asciidoctor-url: http://asciidoctor.org
11 :asciidoctorj-url: https://github.com/asciidoctor/asciidoctorj
12 :asciidoctorjs-url: https://github.com/asciidoctor/asciidoctor.js
13 :wildfly-url: http://download.jboss.org/wildfly/8.0.0.CR1/wildFly-8.0.0.CR1.zip
14
15 Try the editor online at \(online-demo\) (running on OpenShift).
16
17 This project gives you the possibility to work on the same AsciiDoc file with a team.
18 It's based on the \(asciidoctor-url\) [Asciidoctor project] thanks to :
19
20 * the Java API provided by \(asciidoctorj-url\) [Asciidoctorj]
21 * the Javascript libraries provided by \(asciidoctorjs-url\) [Asciidoctor.js]
22
23 [[drag-drop]]
24 .Drag and drop feature
25 !image::ad-editor-dragdrop.png[Drag and drop feature]
26
27 [[offline]]
28 .Offline mode feature
29 !image::ad-editor-offline.png[Offline mode feature]
30
31
32 == 0.1.0-alpha3
33
34 * *Full offline mode* : if you are not connected to the server by WebSocket protocol,
35 rendering view is still working in real-time with the Javascript project asciidoctor,
36
37
38 *
39 == 0.1.0-alpha2 features
40
41
```

**Right Panel (Rendered HTML):**

Real-time collaborative editor for  
Asciidoc files

Try the editor online at <http://wildfly-mgreau.rhcloud.com/ad-editor> (running on OpenShift).

This project gives you the possibility to **work on the same AsciiDoc file with a team** and see the rendering in realtime.

It's based on the [Asciidoctor project](#) thanks to :

- the Java API provided by [Asciidoctor](#)
- the Javascript libraries provided by [Asciidoctor.js](#)

Figure 1. Drag and drop feature

Figure 2. Offline mode feature

0.1.0-alpha3

# Asciidoctor resources

**Website** <http://asciidoctor.org> (blog, user manual, writing guide...)

**Github** <http://github.com/asciidoctor>

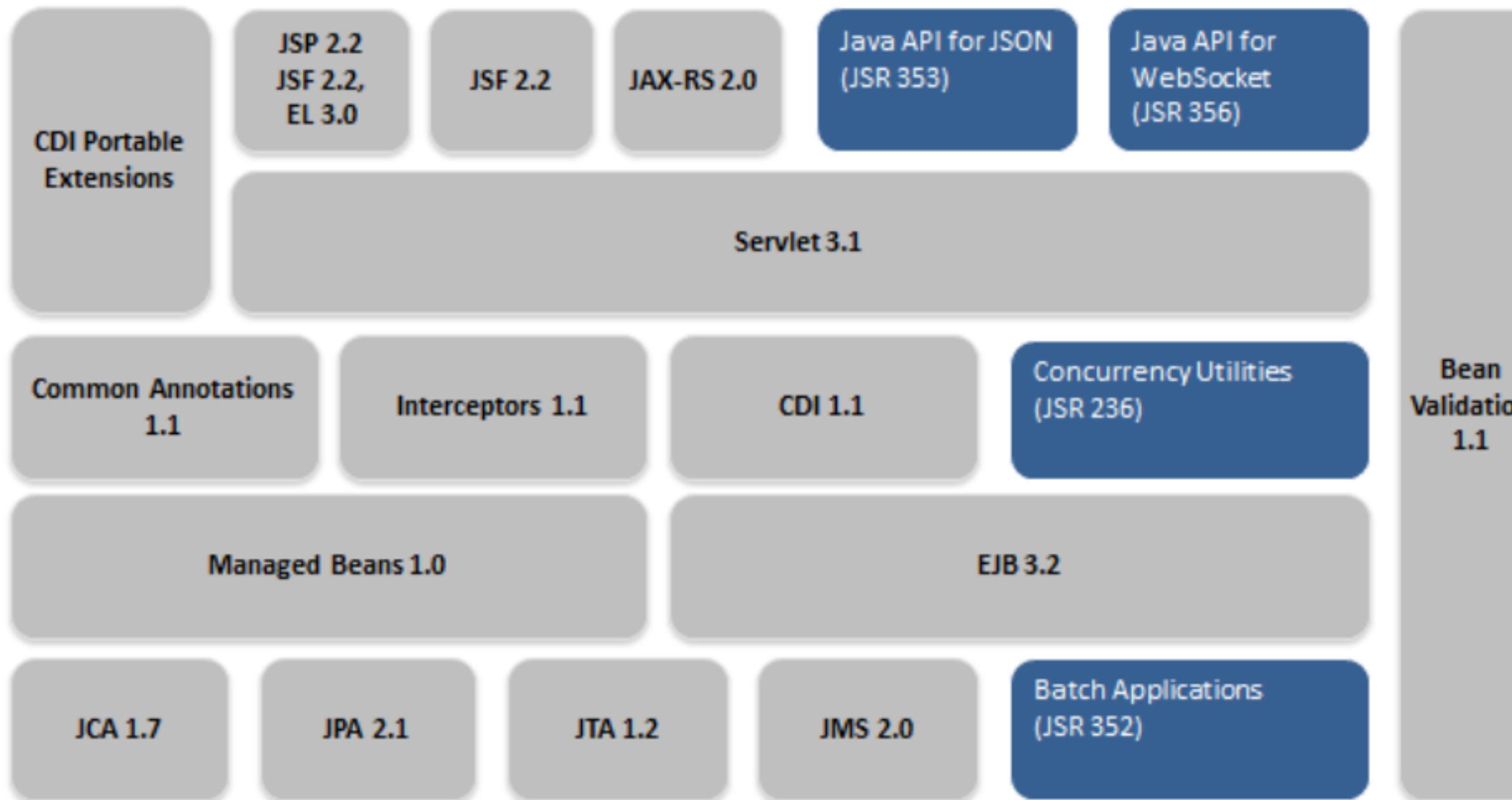
**Forum** <http://discuss.asciidoctor.org>

**Twitter** [@asciidoctor](https://twitter.com/asciidoctor) [@mojavelinux](https://twitter.com/mojavelinux)

[@alexstob](https://twitter.com/alexstob) [@lightguardjp](https://twitter.com/lightguardjp)

# Java EE 7 WebSocket

# Java EE 7 - Overview



HTTP

# half-duplex verbose hack for server push

**“**WebSocket is a *full-duplex bi-directional protocol, over a Single TCP Connection.*

— Arun Gupta (RedHat)

# WebSocket

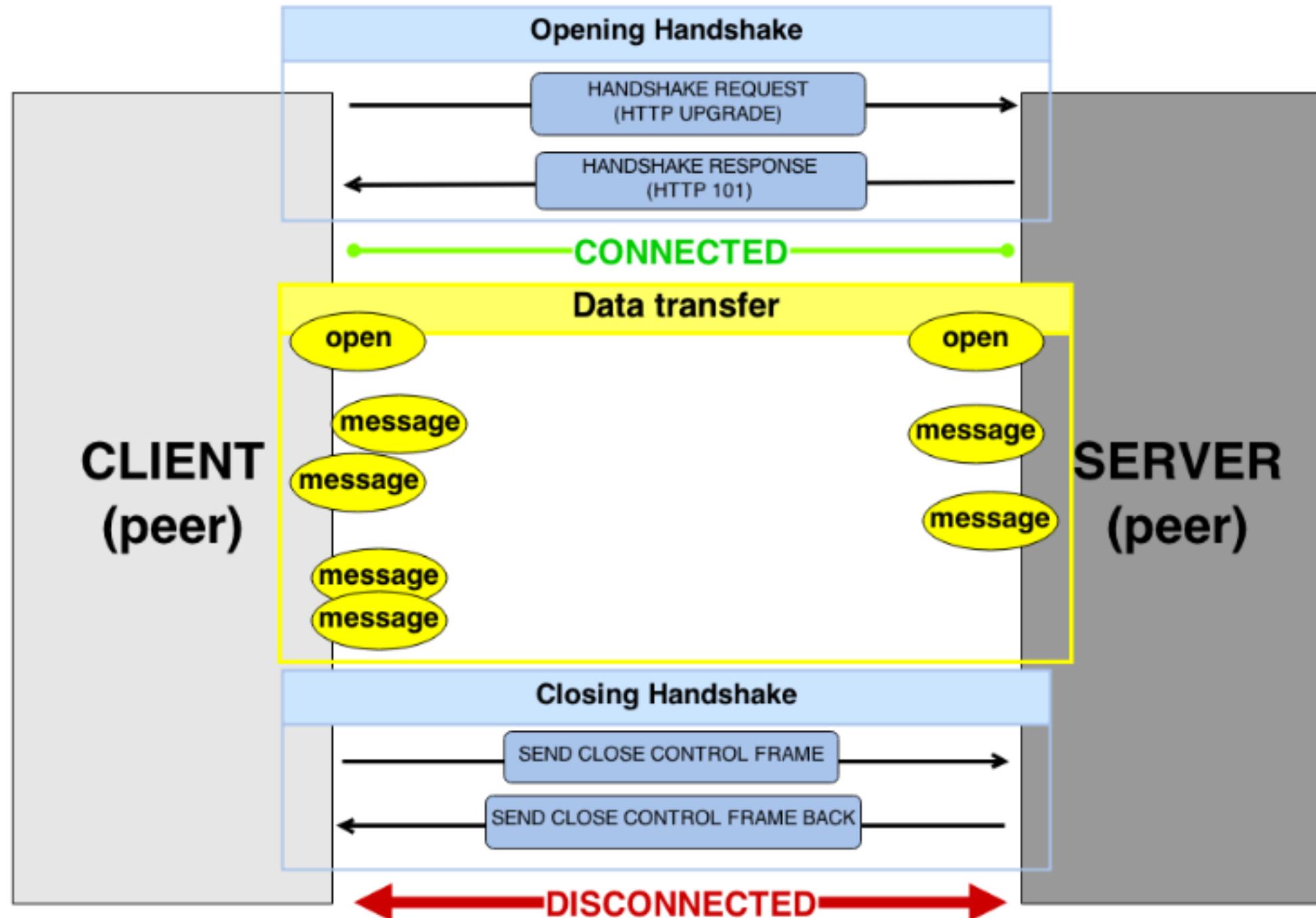
**WebSocket Protocol** IETF -

RFC6455

**Javascript API** W3C

**Java API for WebSocket** JSR 356

# WebSocket Protocol lifecycle



# Request

```
GET /ad-editor/adoc/1234 HTTP/1.1      (1)
Host: wildfly-mgreau.rhcloud.com:8000  (2)
Upgrade: websocket (3)
Connection: Upgrade (4)
Origin: http://wildfly-mgreau.rhcloud.com
Sec-WebSocket-Key: 0EK7XmpTZL341o0h7x1cDw==
Sec-WebSocket-Version: 13
```

# Response

```
HTTP/1.1 101 Switching Protocols
Connection:Upgrade
Sec-WebSocket-Accept:SuQ5/hh0kStSr6oIzDG6gRfTx2I=
Upgrade:websocket
```

## WebSocket Protocol - Handshake

# Javascript WebSocket API example

```
var wsUri = "ws://echo.websocket.org/";
function testWebSocket() {
    websocket = new WebSocket(wsUri);
    websocket.onopen = function(evt) { onOpen(evt) };
    websocket.onclose = function(evt) { onClose(evt) };
    websocket.onmessage = function(evt) { onMessage(evt) };
    websocket.onerror = function(evt) { onError(evt) };
}
function onOpen(evt) {
    writeToScreen("CONNECTED");
    doSend("WebSocket rocks");
}
function onClose(evt) {
    writeToScreen("DISCONNECTED");
}
function onMessage(evt) {
    writeToScreen('<span>RESPONSE: ' + evt.data+ '</span>');
    websocket.close();
}
```

# Java API for WebSocket - JSR356

## Server and Client Endpoint

**Annotated** `@ServerEndpoint`

`@ClientEndpoint`

**Programmatic** `Endpoint`

## Send and consume messages

**all types** `text, binary or control msg`

**as Java Objects** `Encoders Decoders`

**send synchronously / asynchronously**

# Java API - JSR356 - @ServerEndpoint

## EchoServer.java

```
import javax.websocket.OnMessage;
import javax.websocket.ServerEndpoint;

@ServerEndpoint("/echo")
public class EchoServer {

    @OnMessage
    public String handleMessage(String message){
        return "Thanks for the message: " + message;
    }
}
```

# Java API - JSR356 - Annotations

**@ServerEndpoint** POJO → Server Endpoint

**@ClientEndpoint** POJO → Client Endpoint

**@OnOpen** Intercepts open events

**@OnMessage** Intercepts message events

**@OnClose** Intercepts close events

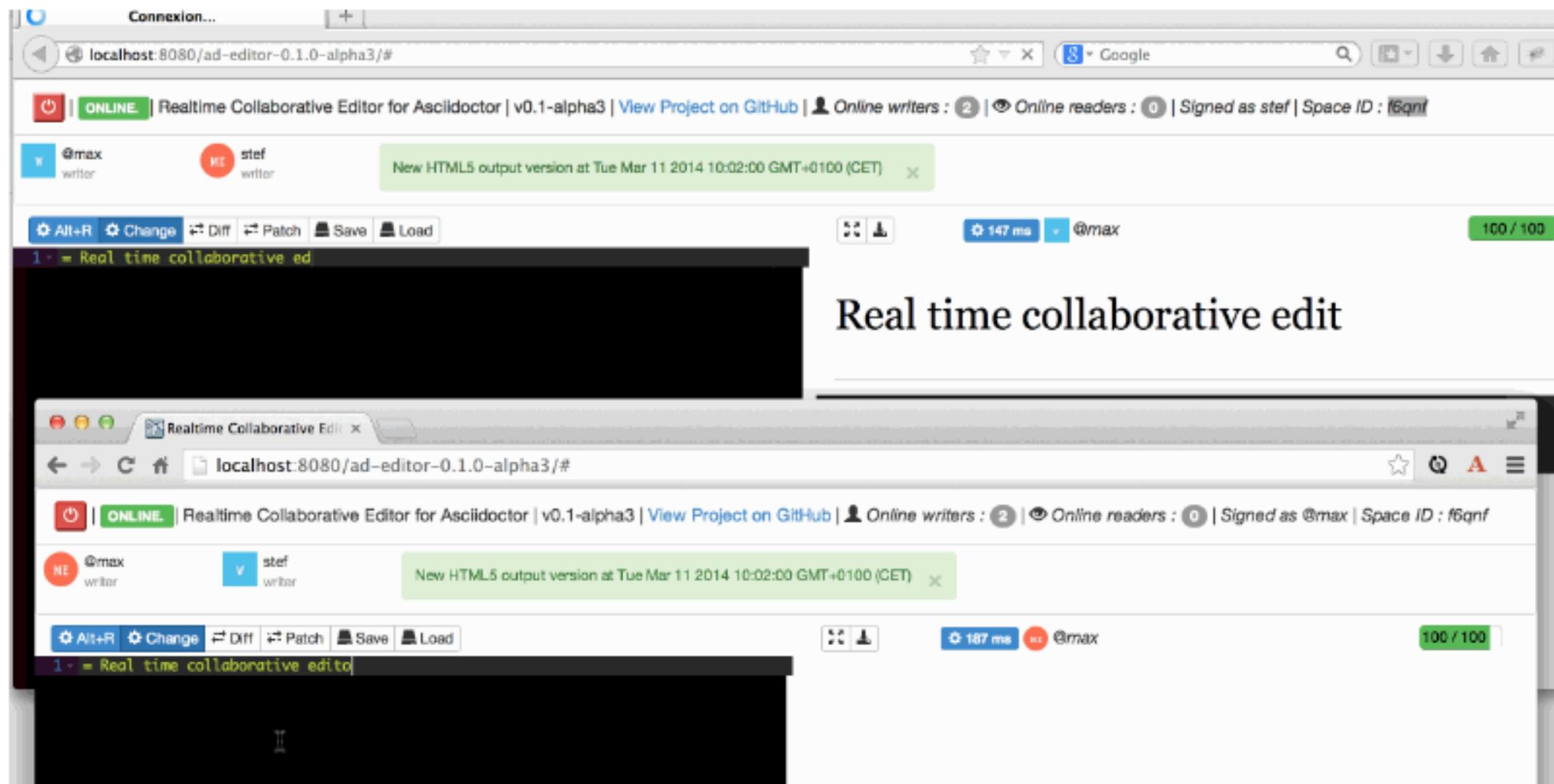
**@OnError** Intercepts error events

# ad-editor Demo & Code

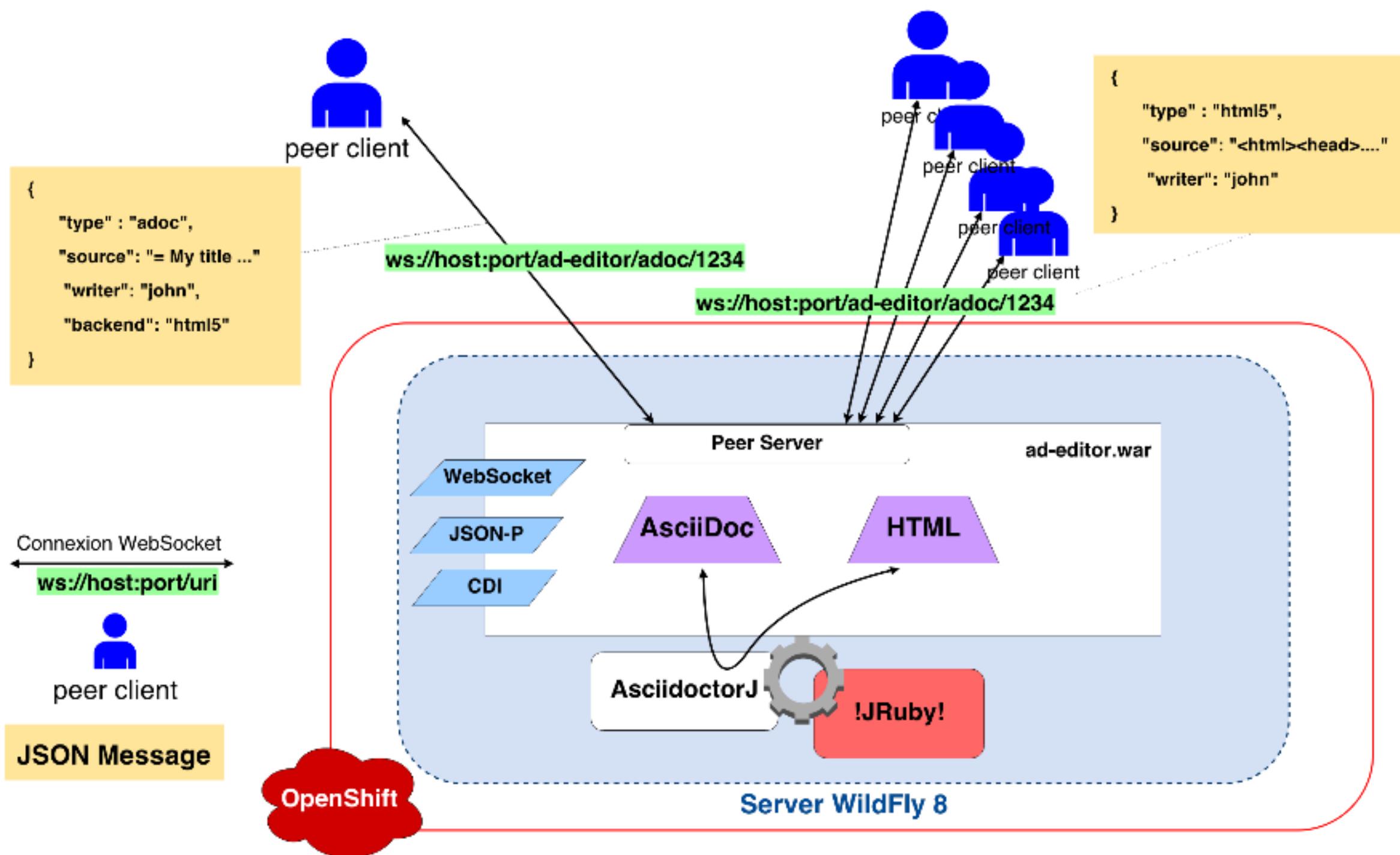
# Asciidoctor via asciidoctorJ and WebSocket

**http://tinyurl.com/adeditor #adeditor**

**DEMO TIME**



# How does ad-editor work ?



## ad-editor : API Javascript send Message → Java API

### services.js

```
app.factory('WebSocketService', function($window) {
  ...
  // Send an adoc source to see the generated output back
  service.sendAdocSource = function(idAdoc, source, writer, backend) {
    var wsUrl;
    ...
    var websocket = new WebSocket(wsUrl);
    var jsonObj = {"type" : backend, "source" : source, "writer": writer};
    websocket.send(JSON.stringify(jsonObj));
  };
  ...
});
```

## ad-editor : Java API @ServerEndpoint

```
@ServerEndpoint(value = "/adoc/{adoc-id}", (1)
    decoders = { MessageDecoder.class }, (2)
    encoders = { AsciidocMessageEncoder.class ... }) (3)
public class WWSMADEndpoint {

    static Set<Session> peers = Collections (4)
        .synchronizedSet(new HashSet<Session>());
    @Inject @Backend("html5")
    Event<AsciidocMessageEvent> html5Event; (5)

    @OnMessage
    public void message(AsciidocMessage msg, Session session, (6)
                           @PathParam("adoc-id") String adocId) {
        AsciidocMessageEvent event =
            new AsciidocMessageEvent(session, adocId, msg);
        ...
        html5Event.fire(event); (7)
    }
}
```

### AsciidocMessageConsumer.java

```
public class AsciidocMessageConsumer {  
  
    @Inject  
    AsciidoctorProcessor processor; (1)  
  
    public void html5RenderedEvent(@Observes @Backend("html5") (2)  
                                    AsciidocMessageEvent event){  
        OutputMessage html;  
        ...  
        html.setContent(  
            processor.renderAsDocument(event.msg.getAdocSource())); (3)  
  
        // send the new HTML version to all connected peers  
        WWSMADEndpoint.sendMessage(html, event.id); (4)  
    }  
}
```

## ad-editor : API Java send → API Javascript Consume

```
@ServerEndpoint(value = "/adoc/{adoc-id}", (1)
    decoders = { MessageDecoder.class }...
public class WWSMADEndpoint {
    ...
    public static void sendMessage(Message msg, String adocId) {
        for (Session session : peers) {
            ...
            session.getAsyncRemote().sendObject(msg); (3)
        }
    }
}
```

```
app.controller("adEditorCtrl", function($scope, WebSocketService) {
    WebSocketService.subscribe(function(id, message) {
        var obj = JSON.parse(message);
        $scope.space[id].html5 = obj.data;
    });
});
```

# What about a Killer feature ?

# Asciidoctor AST Transformation

# Asciidoctor AST Transformation

Asciidoctor → AST feature...

# Asciidoctor AST Transformation

Asciidoctor → AST feature...  
**easy to render **one part** of the  
document...**

# Asciidoctor AST Transformation

Asciidoctor → AST feature...  
easy to render one part of the  
document...  
and we can render HTML5  
slides...

# Asciidoctor AST Transformation

Asciidoctor → AST feature...  
easy to render one part of the  
document...  
and we can render HTML5  
slides...  
and we have the WebSocket  
protocol...

<http://tinyurl.com/webinar85>

SpaceID : 85

DEMO-TIME

# Start your engine

# **What's next for ad-editor ?**

**Github workflow**

**PDF renderer**

**OAuth**

**Chat (audio, text)**

**... Pull requests are welcome :)**

# Thanks to...

@alexsotob

@tgrall

@arungupta

@mojavelinux

when-websocket-met-asciidoc on Github

Maxime Gréau - @mgreau