



Konfigurieren eines JavaFX™ Projekts als Web-Projekt

mit

JPRO

Java in the Browser

TWS

POS
Informatik

SYP
Informatik

ITP3
IT-Medientechnik

SEW3
IT-Medientechnik

**Mit JPro können JavaFX-
Anwendungen als Webseiten
gehostet werden**

JPro for Java/JavaFX

https://www.jpro.one/?gclid=CjwKCAjw7e_0BRB7EiwAIH-go...

JPRO HOME DEMOS PRICES DOC CONTACT



JPRO

Java in the Browser

Get Started! Hello World

email [Subscribe to our JPro Newsletter](#)

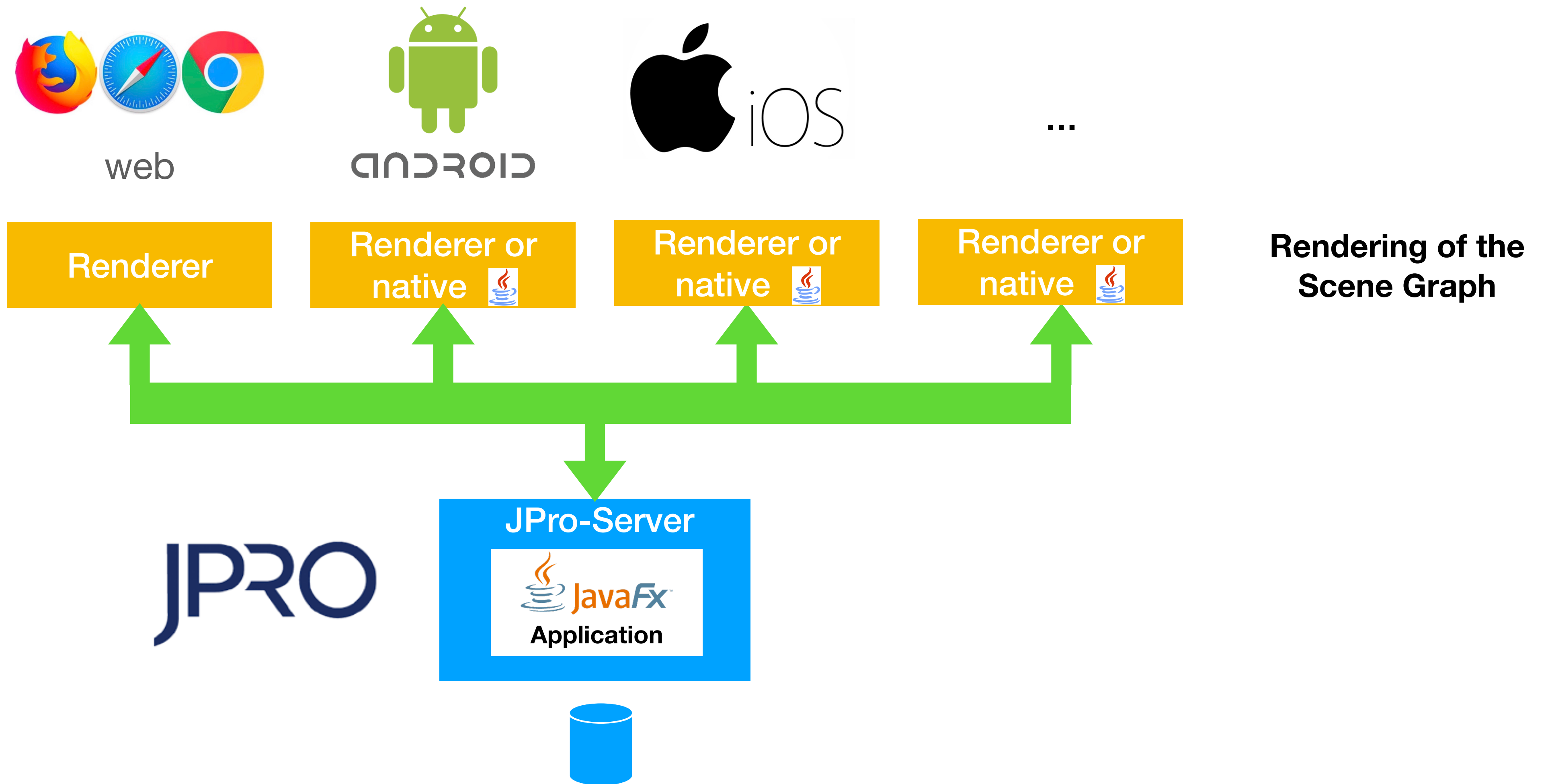
JAVA FOR ALL PLATFORMS!
And, of course, there are NO PLUGINS involved.



JPRO
www.jpro.one

<https://www.jpro.one>


JPro Architecture

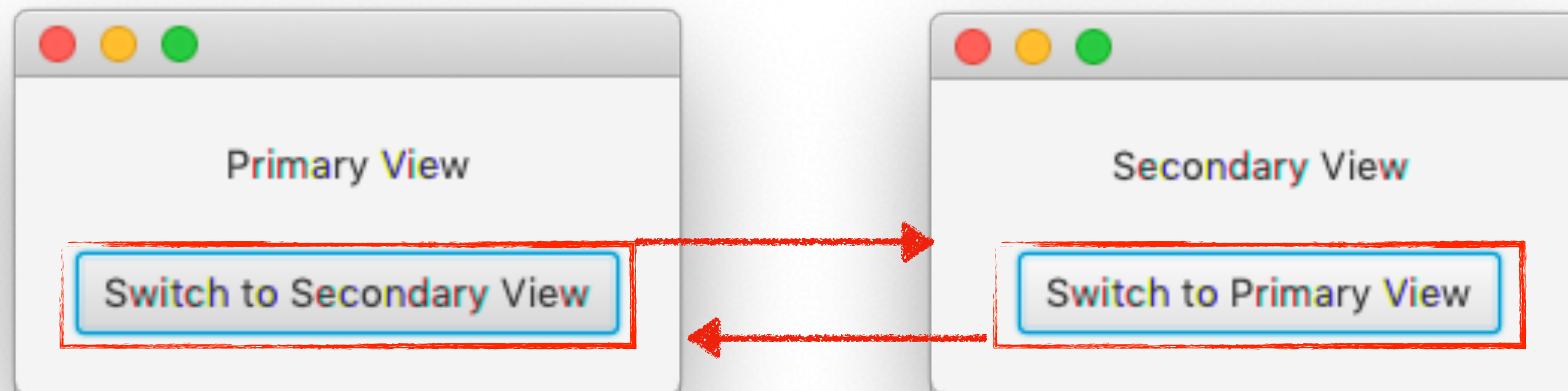


Build Tool

- Grundsätzlich wird **gradle** empfohlen.
- Verwendung von **maven** ebenfalls möglich.

First Application

- Erstellen Sie anhand der Präsentation „05a_JavaFX-create-new-project-modular“ eine erste Anwendung
- Starten Sie die Anwendung: rechte Maustaste auf  App - Run 'App.main()'



Adaptieren der Anwendung für JPro

<https://gist.github.com/htl-leonding/4e420ef1e99b524f927e0944386f4cd5>

Im Programmcode sind KEINE Änderungen durchzuführen

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.
  <modelVersion>4.0.0</modelVersion>

  <groupId>at.htl.demo</groupId>
  <artifactId>javafx-demo</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>

  <properties>
    <jpro.version>2019.2.2</jpro.version>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
    <javafx.version>14.0.1</javafx.version>
  </properties>
  <dependencies>
```

FXML-Dependencies

```
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-controls</artifactId>
  <version>${javafx.version}</version>
  <scope>compile</scope>
</dependency>
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-web</artifactId>
  <version>${javafx.version}</version>
  <scope>compile</scope>
</dependency>
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-swing</artifactId>
  <version>${javafx.version}</version>
  <scope>compile</scope>
</dependency>
```

```
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-fxml</artifactId>
  <version>${javafx.version}</version>
  <scope>compile</scope>
</dependency>
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-media</artifactId>
  <version>${javafx.version}</version>
  <scope>compile</scope>
</dependency>
```

JPro-Dependency

```
<dependency>
  <groupId>com.sandec.jpro</groupId>
  <artifactId>jpro-webapi</artifactId>
  <version>${jpro.version}</version>
  <scope>compile</scope>
</dependency>
```

JPro-maven-plugin

```
</dependencies>
<build>
  <plugins>
    <plugin>
      <groupId>com.sandec.jpro</groupId>
      <artifactId>jpro-maven-plugin</artifactId>
      <version>${jpro.version}</version>
      <configuration>
        <visible>>false</visible>
        <JVMArgs>
          <!-- <JVMArg>your-args</JVMArg> -->
        </JVMArgs>
        <mainClassName>at.htl.demo.App</mainClassName>
        <openingPath></openingPath>
      </configuration>
    </plugin>
  </plugins>
```

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.8.1</version>
  <configuration>
    <release>11</release>
  </configuration>
</plugin>
<plugin>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-maven-plugin</artifactId>
  <version>0.0.4</version>
  <configuration>
    <mainClass>at.htl.demo.App</mainClass>
  </configuration>
</plugin>
</plugins>
</build>
```

jpro-repositories

```
<pluginRepositories>
  <pluginRepository>
    <id>jpro - sandec repository</id>
    <url>https://sandec.bintray.com/repo</url>
  </pluginRepository>
</pluginRepositories>
<repositories>
  <repository>
    <id>jpro - sandec repository</id>
    <url>https://sandec.bintray.com/repo</url>
  </repository>
</repositories>
```

</project>

Starten der Anwendung

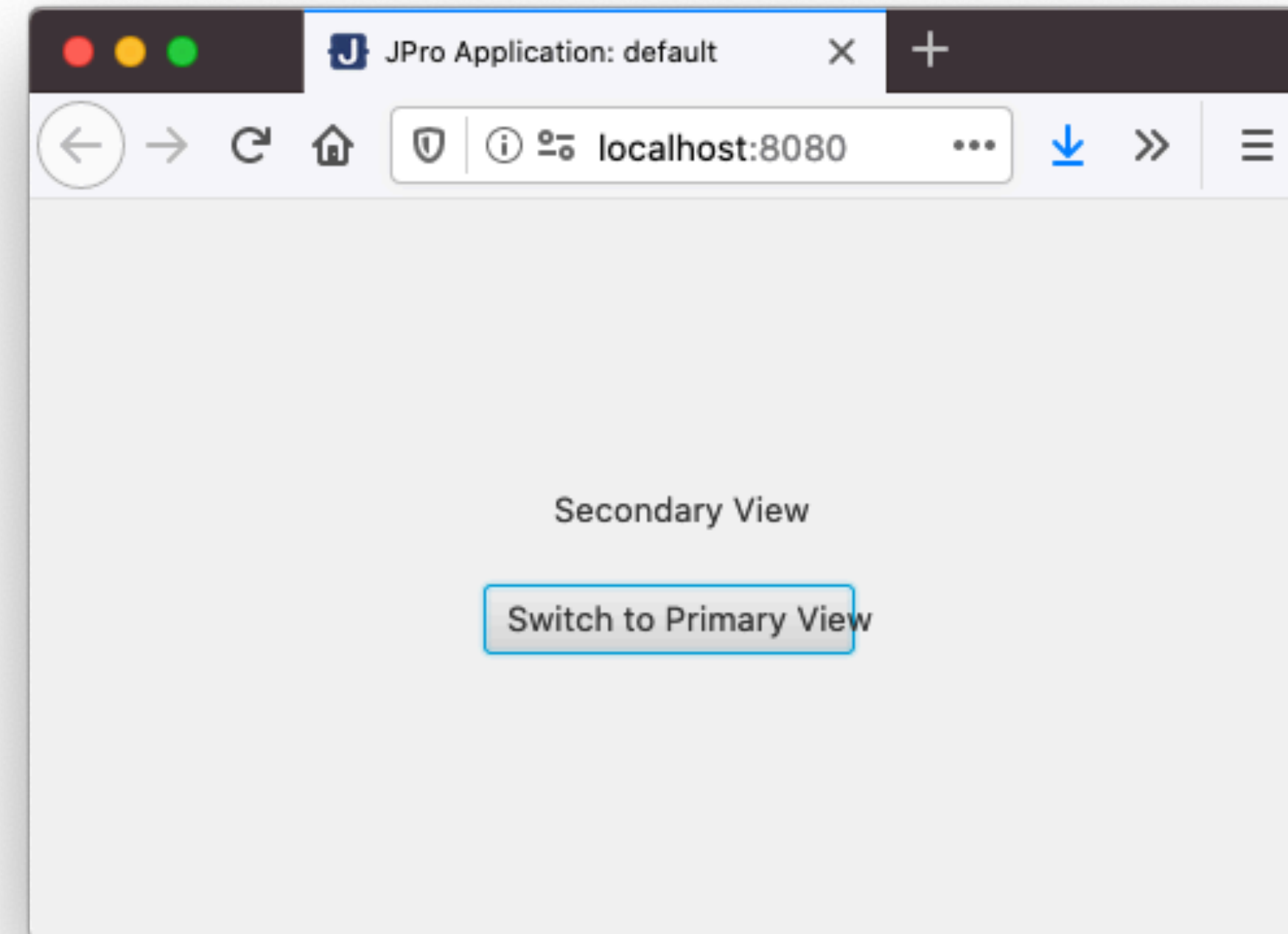
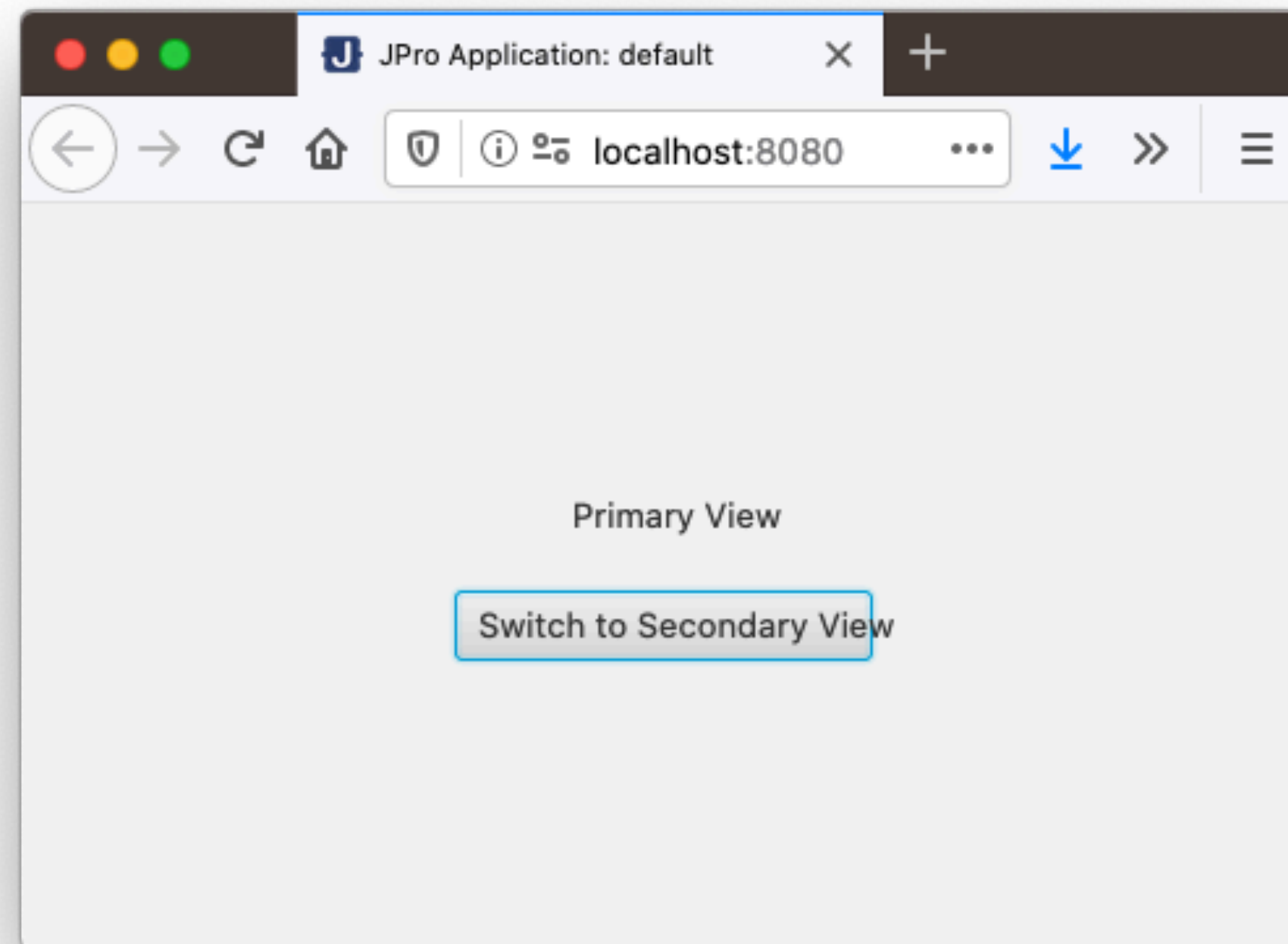
The screenshot shows an IDE window titled "javafx-demo - App.java". The central editor displays the following code:

```
1 package at.htl.demo;
2
3 import ...
4
11
12 public class App extends Application
13
14     private static Scene scene;
15
16     @Override
17     public void start(Stage stage) throws Exception {
18         scene = new Scene(loadFXML("primary"));
19         stage.setScene(scene);
20         stage.show();
21     }
22
23     static void setRoot(String fxml) {
24         scene.setRoot(loadFXML(fxml));
25     }
26
27     private static Parent loadFXML(String fxml) {
28         FXMLLoader fxmlLoader = new FXMLLoader(App.class.getResource(fxml + ".fxml"));
29         return fxmlLoader.load();
30     }
31
32 public static void main(String[] args) {
33     launch(args);
34 }
35
```

The Maven tool window on the right shows the "jpro:run" goal highlighted in red. A context menu is open over it, showing options like "Run Maven Build", "Run 'Unnamed'", "Debug 'Unnamed'", "Create 'Unnamed'...", "Execute Before Build", "Execute After Build", "Execute Before Rebuild", "Execute After Rebuild", "Execute Before Run/Debug...", and "Assign Shortcut...".

At the bottom right, a blue callout box contains the text: "oder in der Console: mvn jpro:run".

Output im Browser



Lizenzen

leider kein
open-source-
projekt

https://www.jpro.one/?page=prices

All the categories listed here are subject to **free JPro licences:**

- Non-Commercial**
The non-commercial use of JPro is free of charge. As soon as JPro is used for direct or in-direct commercial use, the JPro licence prices listed above apply.
- Education**
Universities, schools and similar institutions are welcome to use JPro free of charge for educational purposes.
- Development Servers**
It is common to setup dedicated **development** and **test environments** for software solutions. JPro licences are NOT required for that purpose. The same goes for staging environments. JPro licences are required for live servers dedicated for commercial use, only.
- Development**
As long as you develop and test your software using JPro, no licences for that software are required. JPro licences are **not** required during the development and testing phase of a software project. As soon as, but not before, a solution is turned into commercial use, JPro licences are required.
- Open Source**
As long as a project comply with a common Open Source licence agreement, it is welcome to use JPro free of charge.
- Staging Servers**
It is common to setup dedicated **staging environments** for software solutions. JPro licences are NOT required for that purpose. The same goes for development and testing environments. JPro licences are required for live servers dedicated for commercial use, only.

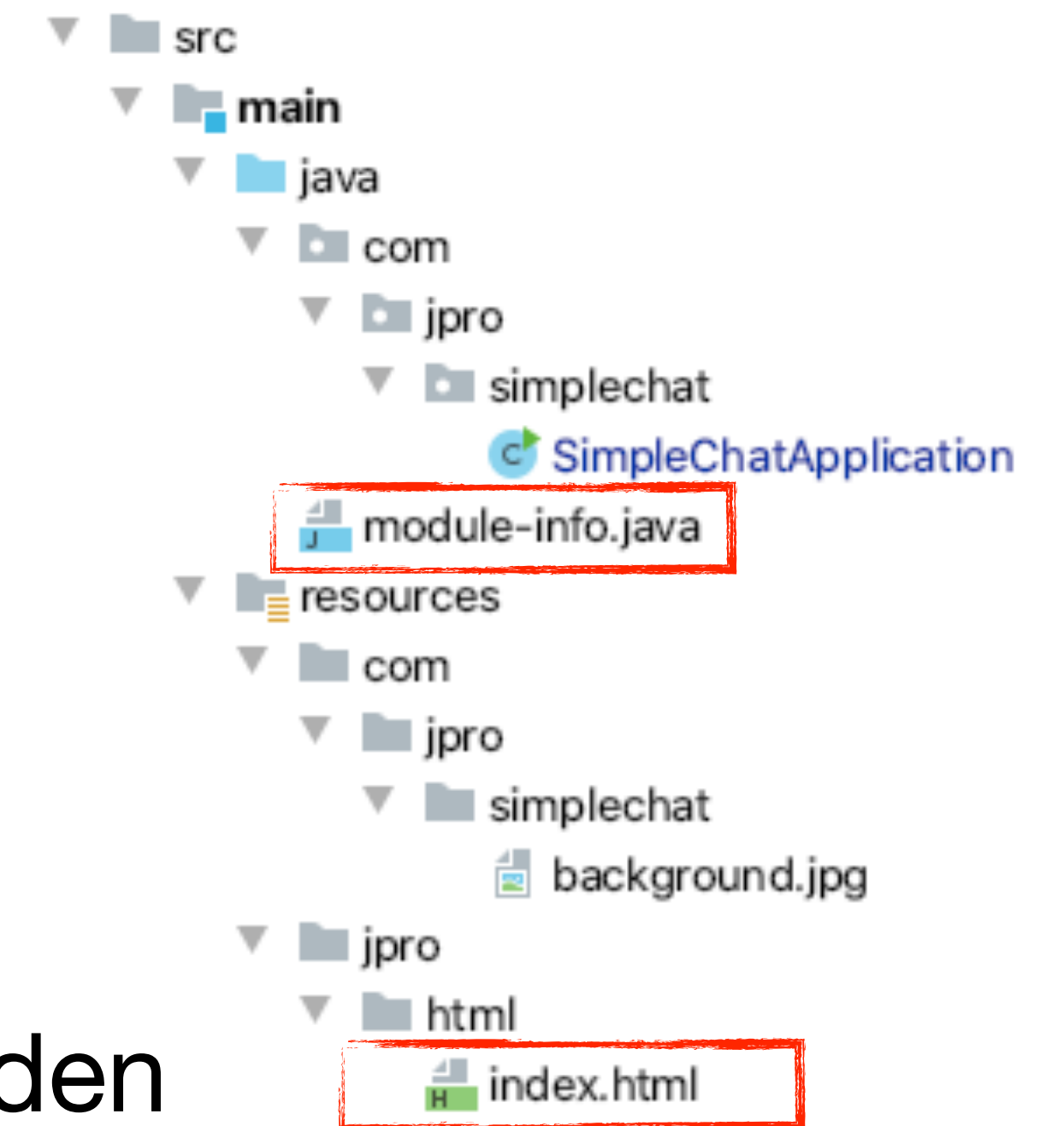
Ein zweites Projekt

JProSimpleChat

<https://github.com/htl-leonding/jpro-javafx-gradle-simple-chat>

Technische Daten

- In Gradle geschrieben
- verwendet modulares Java
- Nachrichten werden in einer ListView gespeichert
- Die JavaFX-Applikation wird in ein html-Seite eingebunden



build.gradle

- hat die gleiche Funktion wie pom.xml
- verwendet die Programmiersprache Groovy als „gradle build language“
- es könnte auch Kotlin als „gradle build language“ verwendet werden
- Man konfiguriert
 - die Repositories
 - Java
 - JPro
- <https://bit.ly/34LvEbC>

```

public class SimpleChatApplication extends Application {

    static public ObservableList<String> messages = FXCollections.observableArrayList();

    @Override
    public void start(Stage stage) throws Exception {

        ListView<String> messagesView = new ListView<String>(messages);
        TextField input = new TextField();
        input.setPromptText("Your Message");
        Button button = new Button("send");
        HBox.setHgrow(input, Priority.ALWAYS);

        VBox vbox = new VBox(messagesView, new HBox(input,button));
        vbox.setMaxHeight(500);
        vbox.setMaxWidth(400);

        EventHandler<ActionEvent> handler = (e) -> {
            if(!input.getText().isEmpty()) {
                messages.add(input.getText());
                input.setText("");
            }
        };
        input.setOnAction(handler);
        button.setOnAction(handler);

        StackPane container = new StackPane(vbox);
        container.setAlignment(Pos.CENTER);
        container.setStyle("-fx-background-image: url('/com/jpro/simplechat/background.jpg');" +
            "-fx-background-size: cover;");

        stage.setScene(new Scene(container, 800, 800));
        stage.show();
    }
}

```

index.html

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>jpro Application: SimpleChat</title>
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
```

```
<link rel="stylesheet" type="text/css" href="/jpro/css/jpro.css">
```

```
<script src="/jpro/js/jpro.js" type="text/javascript"></script>
```

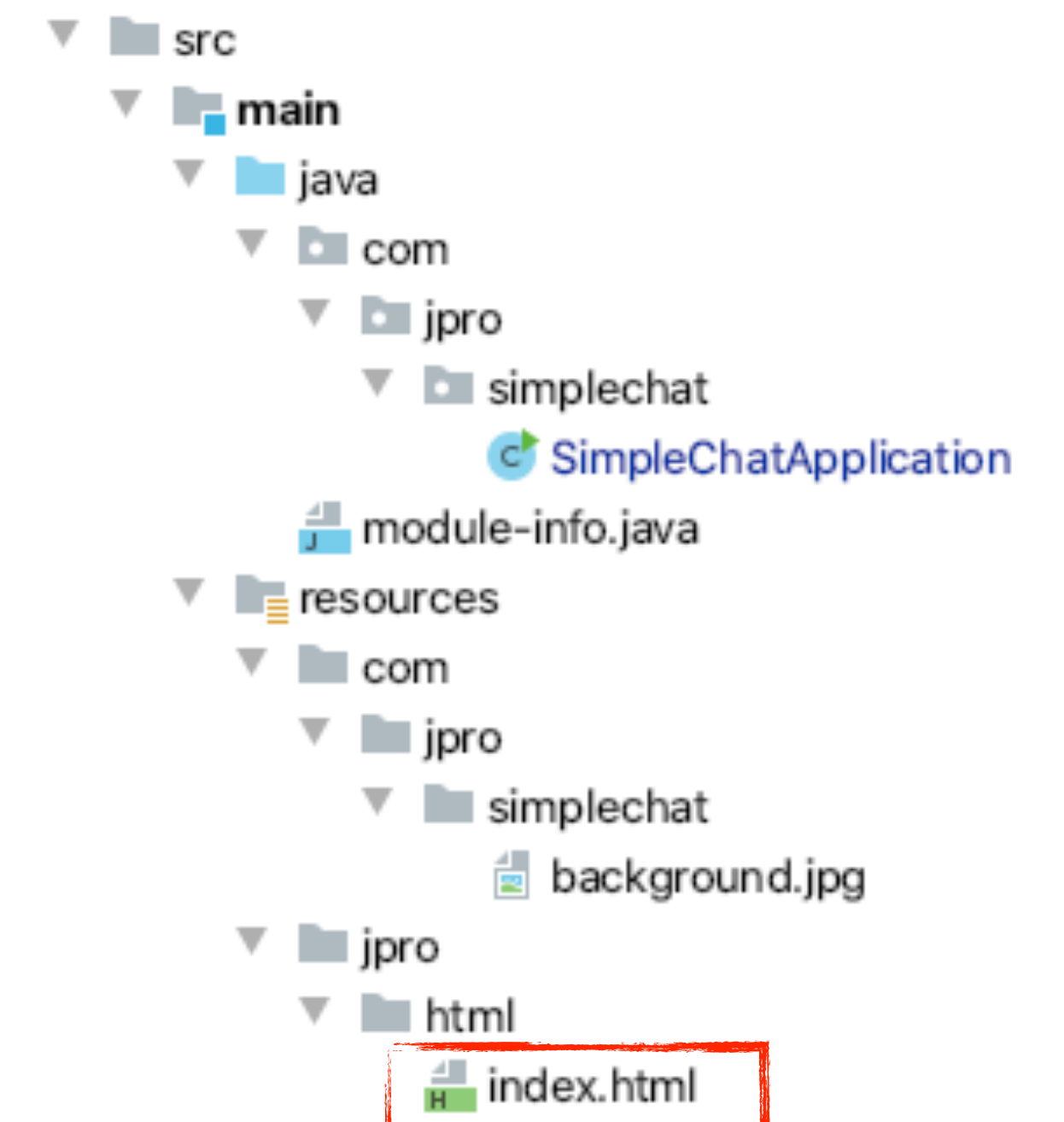
```
</head>
```

```
<body>
```

```
<jpro-app href="/app/default" fullscreen="true"></jpro-app>
```

```
</body>
```

```
</html>
```



Starten der Anwendung

The screenshot shows an IDE window titled "JProSimpleChat - index.html [JProSimpleChat.main]". The main editor displays the content of "index.html" with the following code:

```
1 <!DOCTYPE html>
2
3 <html>
4 <head>
5   <title>jpro Application: Hello JPro</title>
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7
8   <link rel="stylesheet" type="text/css" href="/jpro/css/jpro.css">
9   <script src="/jpro/js/jpro.js" type="text/javascript"></script>
10
11 </head>
12
13 <body>
14
15   <jpro-app href="/app/default" fullscreen="true"></jpro-app>
16
17 </body>
18
19 </html>
20
21
```

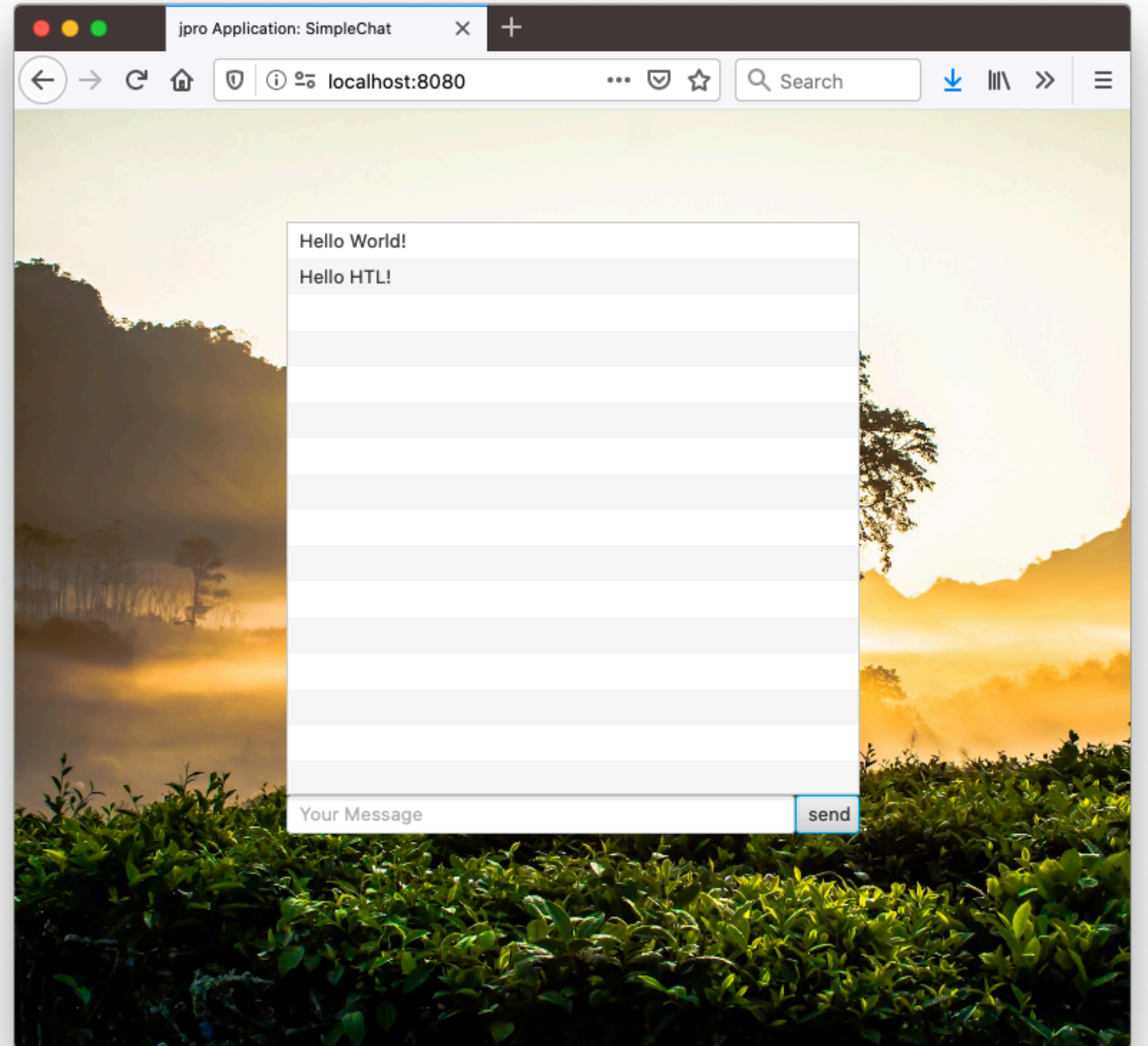
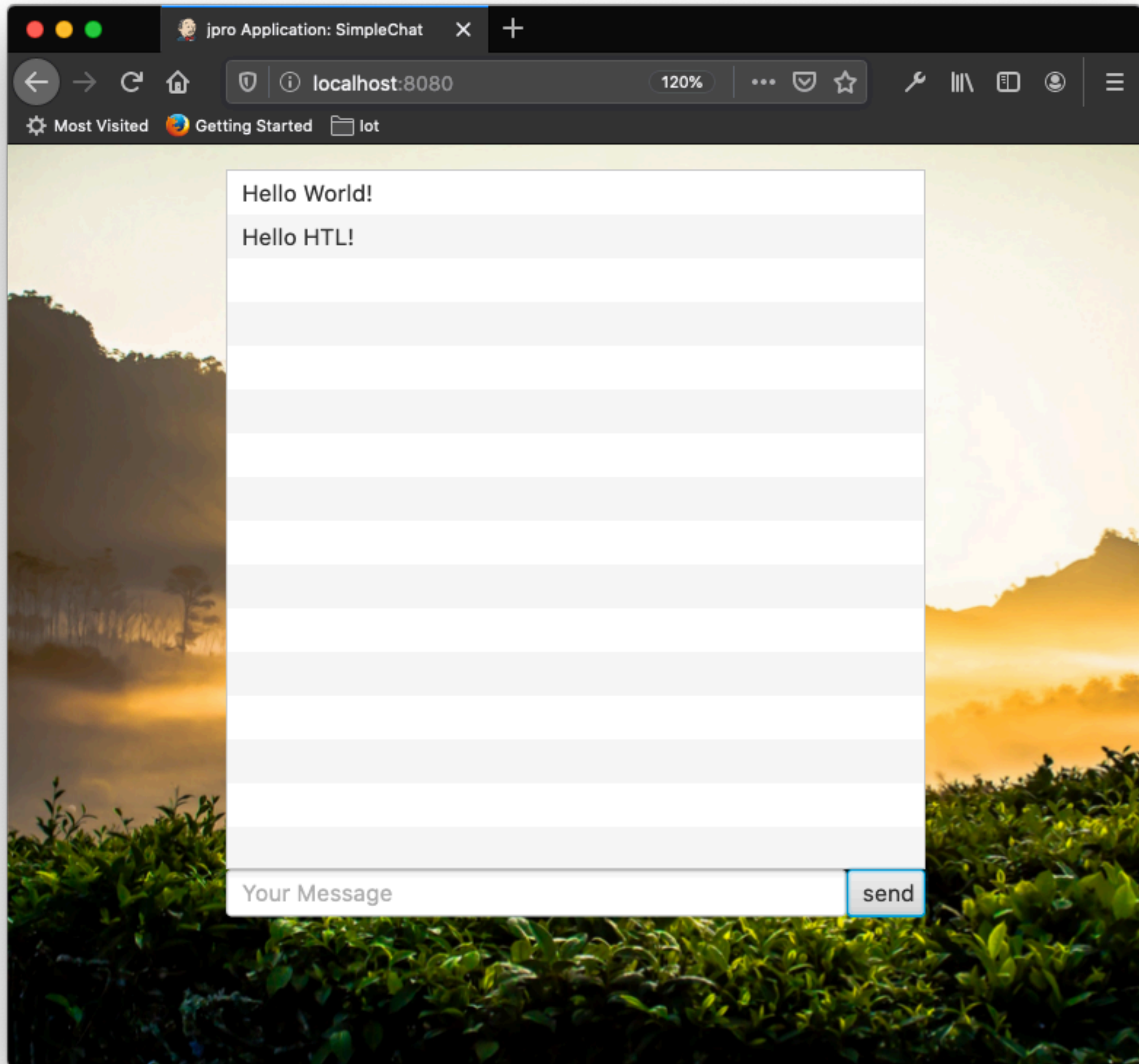
On the right side, the Gradle tool window shows the project structure:

- JProSimpleChat
 - Tasks
 - application
 - build
 - build setup
 - distribution
 - documentation
 - help
 - jpro
 - jproRestart
 - jproRun
 - jproStart**
 - jproStartAs...
 - jproStartCo...
 - jproStop
 - prepareBro...
 - other
 - verification
 - Dependencies
 - Run Configuration

A context menu is open over the "jproStart" task, listing the following options:

- Run 'JProSimpleChat [jpro...]' (^⇧R)
- Debug 'JProSimpleChat [jpro...]' (^⇧D)
- Run 'JProSimpleChat [jpro...]' with Coverage
- Create 'JProSimpleChat [jpro...]'...
- Jump to Source (⌘⇧↓)
- Execute Before Sync
- Execute After Sync
- Execute Before Build
- Execute After Build
- Execute Before Rebuild
- Execute After Rebuild
- Assign Shortcut...

The IDE interface includes a Project view on the left, a Commit view, and a Favorites view. The bottom status bar shows "4: Run", "8: Services", "Terminal", "9: Git", "6: TODO", and "Event Log".



Nachteile von Pro

Not yet supported JavaFX features

(this list refers to version 2019.2.2)

- MediaPlayer
- The following Effect classes
 - Bloom
 - BoxBlur
 - ColorInput
 - DisplacementMap
 - DropShadow (Supported, but slow)
 - Glow
 - ImageInput
 - InnerShadow (Supported, but slow)
 - Lighting
 - MotionBlur
 - PerspectiveTransform
 - Reflection
 - Shadow
- 3D
- SwingNode
- javafx.stage.FileChooser, please take a look at our [WebAPI](#) and our [sample project](#) for an alternative.
- Clipboard
- WebView (Supported in it's basic form, only. We recommend to use the [HTMLView](#), instead.)

Sources

- <https://github.com/JPro-one/JProSimpleChat/blob/master/src/main/java/com/jpro/simplechat/SimpleChatApplication.java>
- <https://www.youtube.com/watch?v=xvEhp-1kyWg>



Noch
Fragen?

HTL LE  NDING

Schön, hier zu lernen