ＳＴＥＰ0１

Ｓｐｒｉｎｇフレームワーク入門

コードで理解するＳｐｒｉｎｇの世界

開発環境構築 ＋ 『Hello World』

目次

[1 ソフトウェアのインストール 1](#_Toc428185711)

[1.1 ソフトウェア構成 1](#_Toc428185712)

[1.2 ダウンロード先 1](#_Toc428185713)

[1.3 その他の要件 1](#_Toc428185714)

[1.4 JDKのインストール 2](#_Toc428185715)

[1.5 Eclipse インストール ＆ 初期設定 2](#_Toc428185716)

[1.6 Tomcat インストール 3](#_Toc428185717)

[1.7 Eclipse開発環境 ／ Tomcat Local Server セットアップ 3](#_Toc428185718)

[1.8 Eclipse開発環境 ／ CSS、HTML、JSPファイルの文字コード設定の確認と変更 3](#_Toc428185719)

[1.9 Spring IDE （Eclipseプラグイン）導入 4](#_Toc428185720)

[1.10 Maven設定追加（AT-NET時のみPROXY設定） 4](#_Toc428185721)

[1.11 PostgreSQLインストール 5](#_Toc428185722)

[1.12 PostgreSQLデータベース設定情報 5](#_Toc428185723)

[2 Springを使用したプロジェクト作成 6](#_Toc428185724)

[2.1 Mavenプロジェクト作成 6](#_Toc428185726)

[2.2 プロジェクトファセット変更 6](#_Toc428185727)

[2.3 サーブレット設定ファイル web.xml を作成 7](#_Toc428185728)

[2.4 Maven設定ファイル pom.xml に設定を追加 8](#_Toc428185729)

[2.5 Spring定義ファイル（その１） application-context-biz.xml を作成 9](#_Toc428185730)

[2.6 Spring定義ファイル（その２） application-context-web.xml を作成 10](#_Toc428185731)

[3 Hello Spring アプリケーション作成 11](#_Toc428185732)

[3.1 Hello Spring 画面コントロールクラス作成 11](#_Toc428185734)

[3.2 Hello Spring 画面View作成 11](#_Toc428185735)

[3.3 Hello World 画面表示テスト 12](#_Toc428185736)

# ソフトウェアのインストール

## ソフトウェア構成

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | カテゴリ | 製品名 | バージョン | 備考 |
| 1 | Java | Java SE Development Kit | 1.8.0\_111 | JSDK 1.8系の最新版でOK |
| 2 | 開発環境 | Eclipse IDE for Java EE Developers | 4.6.1 (Neon) | Eclipse Java EE 4.6系の最新版でOK |
| 3 | Spring IDE 3.7.3 RELEASE | 3.8.2 RELEASE |  |
| 4 | junit | 4.12 |  |
| 5 | Spring | Spring Framework | 4.2.8.RELEASE | Spring 4.2系であれば恐らく大丈夫？ |
| 6 | Webｻｰﾊﾞ | Tomcat | 8.5.5 | Tomcat 8.5系の最新版でOK |
| 7 | ﾃﾞｰﾀﾍﾞｰｽ | PostgreSQL | 9.6.0 | PostgreSQL 9.6系 の最新版でOK |

本講習における Spring Framework をはじめとした各ライブラリの構成は、

Spring IO 2.0.8-Release に準拠する。

## ダウンロード先

個別にDLが必要な物だけを記載

|  |  |  |
| --- | --- | --- |
| # | 製品名 | URL |
| 1 | Java SE Development Kit | http://www.oracle.com/technetwork/java/javase/downloads/index.html |
| 2 | Eclipse IDE for Java EE Developers | https://www.eclipse.org/downloads/ |
| 6 | Tomcat | http://tomcat.apache.org/download-80.cgi |
| 7 | PostgreSQL | http://www.postgresql.org/download/ |

## その他の要件

・インターネットから各種パッケージを探しに行くので、インターネット接続が必須

## JDKのインストール

|  |
| --- |
| ① インストール  以下のサイトからダウンロード ⇒ http://www.oracle.com/technetwork/java/javase/downloads/index.html  JDKを選択し普通にインストール  ② 環境変数の設定後，コマンドプロンプトでJavaが実行できるか確認する．  --- console ---  $ java -version  java version "1.8.0\_111"  Java(TM) SE Runtime Environment (build 1.8.0\_111-b14)  Java HotSpot(TM) Client VM (build 25.111-b14, mixed mode)  ------------- |

## Eclipse インストール ＆ 初期設定

|  |
| --- |
| ① Eclipseのインストール  ここではインストールフォルダを「C:/DEV/ 」とする（本当はどこでも良い）  $HOME=C:/DEV/  $HOME/workspace -> Eclipseのプロジェクトフォルダ    Eclipse 4.6.1(Neon)をダウンロード ⇒ https://www.eclipse.org/downloads/  ダウンロードした[eclipse]フォルダを以下に置く．  $HOME/eclipse  Eclipseが起動することを確認する．  $HOME/eclipse/eclipse.exe  起動時にworkspaceを指定する  $HOME/workspace  ② Default のJRE の設定確認  Window → Preference → Java → Installed JREs  ③ Java Compiler の設定確認  Window → Preference → Java → Compiler → Compiler compliance level : 1.8  ④ Default の文字コードをUTF-8 に変更  Window → Preference → General → Workspace  Text file Encoding→Other: UTF-8  ⑤ 行番号の表示  Window → Preference → general → Editors → Text Editor → Show line numbers をチェック  ⑥ ホワイトスペースの表示  Window → Preference → general → Editors → Text Editor → Show whitespace characers をチェック  ⑦ AT-NET Proxy の設定（AT-NETの場合のみ）  Window - Preferences - General - Network Connections  ・ Active Providerを "Manual" に変更  ・ HTTP、HTTPS の Host、Port、User、Passwordに、AT-ProxyとAT-ID、Password を設定 (SOCKSは設定しない)  ⑧ 日本語化（必要に応じて、今回は英語のまま行います） |

## Tomcat インストール

|  |
| --- |
| ① Tomcatインストール  ここではインストールフォルダを「C:/DEV/ 」とする（本当はどこでも良い）  $HOME=C:/DEV/  ダウンロード ⇒ http://tomcat.apache.org/download-80.cgi  Windowsなら、32-bit Windows zip または 64-bit Windows zip を選択。  ダウンロードして解凍  $HOME/tomcat （本当はどこでもよい） |

## Eclipse開発環境 ／ Tomcat Local Server セットアップ

|  |
| --- |
| ① Tomcat Localhost サーバ の作成  Window → Preference → Server → Runtime Environments → [Add]    ② [New Server Runtime Environment]画面で以下を設定して[Next]を押下．  Apache Tomcat v8.5  [Create a new local server]にチェック（このチェックを忘れないように）  [Next]ボタン押下  ③ [Tomcat Server]画面で以下のパラメータを投入する．  --- Parameter ---  Name: Apache Tomcat v8.5  directory: $HOME/tomcat/apache-tomcat-8.5.5  JRE: workbench default JRE  ---------------  [Finish]ボタン押下  ③ Serversビューの表示  Window → Show View → Other → Server → Servers  下部に Servers View が表示されるので、[Tomcat v8.5 Server at localhost] があることを確認 |

## Eclipse開発環境 ／ CSS、HTML、JSPファイルの文字コード設定の確認と変更

|  |
| --- |
| ① CSS ファイルの Default の文字コードが UTF-8 であることを確認、もし違っていれば UTF-8 に変更  Window → Preference → Web → CSS Files 画面 → Creating files → Encoding: → ISO 10646/Unicoede(UTF-8)  ② HTML ファイルの Default の文字コードが UTF-8 であることを確認、もし違っていれば UTF-8 に変更  Window → Preference → Web → HTML Files 画面 → Creating files → Encoding: → ISO 10646/Unicoede(UTF-8)  ③ JSP ファイルの Default の文字コードをUTF-8 に変更  Window → Preference → Web → JSP Files 画面 → Creating files → Encoding: → ISO 10646/Unicoede(UTF-8) |

## Eclipse開発環境 ／ JSPファイルのテンプレートを HTML5 にしてしまう

|  |
| --- |
| ① Window → Preference → Web → JSP Files 画面→ Editor → Templates → New JSP File(html) をダブルクリック  ② Pattern を以下のように編集 |
| <%@ page language="java" contentType="text/html; charset=${encoding}" pageEncoding="${encoding}"%>  <!DOCTYPE html>  <html>  <head>  <meta charset="${encoding}">  <title>Insert title here</title>  </head>  <body>  ${cursor}  </body>  </html> |

## Spring IDE （Eclipseプラグイン）導入

|  |
| --- |
| ① Spring Tool Suite のインストール  Help → Eclipse MarketPlace  検索画面で[Spring]を検索して，以下のツールをインストールする。  -----------------  Spring IDE 3.8.2.RELEASE  -----------------  プラグインはデフォルト選択のままでインストール。インストール後Eclipseが再起動する。 |

## Maven設定追加（AT-NET時のみPROXY設定）

|  |
| --- |
| ① Window → Preference → Maven→ User Settings  User Settings に、[UserDirectory]\.m2\settings.xml を設定  ② [UserDirectory]\.m2\settings.xmlを作成する  ③ 以下の記述を追加 |
| [UserDirectory]\.m2\settings.xml |
| <settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0  http://maven.apache.org/xsd/settings-1.0.0.xsd">  <proxies>  <proxy>  <active>true</active>  <protocol>http</protocol>  <host>atproxy.bb.ntt-at.co.jp</host>  <port>3128</port>  <username>[XXXX 自分のユーザ XXXX]</username>  <password>[XXXX 自分のパスワード XXXX]</password>  <nonProxyHosts></nonProxyHosts>  </proxy>  </proxies>  </settings> |

##　参考 settings.xml：<https://maven.apache.org/settings.html>

## PostgreSQLインストール

|  |
| --- |
| ① インストール  以下のサイトから Installer version Version 9.6.0 (9.6 の最新版でOK)  http://www.enterprisedb.com/products-services-training/pgdownload  Win x86-32 もしくは x86-64を選択し普通にインストール  今回の管理者パスワードは： nttat123 とすること！  上記以外全てデフォルトで良い。  最後の「Stack Builder may be…」のチェックは外してインストールを完了させる。  ② データベースの設定  PostgreSQLのSQL SHELLを起動してPostgreSQLに接続する。  以下のSQLを発行して新規データベースを作成する．  --- Parameter ---  CREATE DATABASE springdb encoding 'UTF8';  ---------------  ## Windows版のPostgreSQLにはpgAdminⅢ というグラフィカルなツールが添付されており、これを利用して  ## データベースの各種管理を行うことも可能である。  ③ 研修用サンプルアプリで使用するテーブルの作成  以下のファイルの定義された CREATE TABLE 文やINSERT文などを実行して、書籍テーブルを作成する．  create\_table.sql |

## PostgreSQLデータベース設定情報

PostgreSQLの接続情報

|  |  |
| --- | --- |
| 管理ユーザ | postgres |
| パスワード | nttat123 |
| データベース | springdb |

書籍テーブルについて

|  |  |
| --- | --- |
| テーブル名 | book |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | カラム名 | 型 | 制約 | 備考 |
| 1 | id | SERIAL | PK | SEQUENCE book\_id\_seq を用いて自動付番する |
| 2 | isbn | VARCHAR(20) | NOT NULL |  |
| 3 | name | VARCHAR(64) | NOT NULL |  |
| 4 | price | INTEGER | NOT NULL |  |
| 5 | lastup | TIMESTAMP |  | default CURRENT\_TIMESTAMP |

# Springを使用したプロジェクト作成



## Mavenプロジェクト作成

|  |
| --- |
| ① EclipseのPackage Explorerタブでプロジェクト作成を選択  File → New → Project → Maven → Maven Project → Next  ② New Maven project - Selectproject name and location画面  --- Parameter ---  「Create a simple project」 を選択  「Use default Workstpace location」 を選択  ---------------  →[ Next]  ③ New Maven project - Selectproject name and location画面  --- Parameter ---  「Group Id:」 jp.sample　＃すべて小文字  「Artifact Id:」 bookmgr　 ＃すべて小文字  「Version:」 0.0.1-SNAPSHOT  「Packaging:」 war  「Name:」 bookmgr  ※他は空白のままでOK  ---------------  →[ Finish]  ④ プロジェクトが作成されプロジェクトがビルドされるのを待つ |

＃ここでワークスペースのビルドが始まる。

＃ビルドの進捗は右下に地味に表示されているだけなので見落とさないこと。

＃ビルドに数分がかかる場合があるが、終わるまで待っているほうが良い。

＃さらに、現段階ではプロジェクトの設定にエラーが検出された状態となるが気にしない。

＃本エラーは、この後の設定で徐々に解消する予定。

## プロジェクトファセット変更

|  |
| --- |
| ① bookmgr プロジェクトを選択して、マウス右ボタンで「Properties」を選択  → Properties ダイアログが表示される  ② Properties ダイアログの左メニューから「Project Facets」を選択  → Project Facets 画面が表示される  ③ Project Facets 画面  --- Parameter ---  「Java」のバージョンを 1.5 から 1.8 に変更  「Dynamic Web Module」のバージョンを 2.5 から 3.0 に変更  ---------------  → [Apply] or [Ok] で設定を保存 |

## サーブレット設定ファイル web.xml を作成

１）web.xml格納フォルダ作成

|  |
| --- |
| /src/main/webapp/WEB-INF/ （新規作成） |
| bookmgr/src/main/webapp で右クリック → New → Folder  ● Folder画面  ---------------  Folder Name: に「WEB-INF」 と入力して Finish  --------------- |

２）web.xmlファイル新規作成

|  |
| --- |
| /src/main/webapp/WEB-INF/web.xml （新規作成） |
| bookmgr/src/main/webapp/ WEB-INF で右クリック → New → File  ● File 画面  ---------------  File Name: に「web.xml」 と入力して Finish  --------------- |

３）web.xmlファイルコーディング

|  |
| --- |
| /src/main/webapp/WEB-INF/web.xml |
| <?xml version="1.0" encoding="UTF-8"?>  <web-app xmlns="http://java.sun.com/xml/ns/javaee"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee  http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"  version="3.0">  <!-- The definition of the Root Spring Container shared by all Servlets and Filters -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>  /WEB-INF/spring/application-context-biz.xml  </param-value>  </context-param>  <!-- Creates the Spring Container shared by all Servlets and Filters -->  <listener>  <listener-class>  org.springframework.web.context.ContextLoaderListener  </listener-class>  </listener>  <!-- Processes application requests -->  <servlet>  <servlet-name>appServlet</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  <init-param>  <param-name>contextConfigLocation</param-name>  <param-value>/WEB-INF/spring/application-context-web.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup>  </servlet>  <servlet-mapping>  <servlet-name>appServlet</servlet-name>  <url-pattern>/</url-pattern>  </servlet-mapping>  </web-app> |

## Maven設定ファイル pom.xml に設定を追加

|  |
| --- |
| /pom.xml |
| <project xmlns="http://maven.apache.org/POM/4.0.0"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  http://maven.apache.org/xsd/maven-4.0.0.xsd">  <modelVersion>4.0.0</modelVersion>  <groupId>jp.sample</groupId>  <artifactId>bookmgr</artifactId>  <version>0.0.1-SNAPSHOT</version>  <packaging>war</packaging>  <name>bookmgr</name>  <properties>  <org.springframework-version>4.2.8.RELEASE</org.springframework-version>  <org.slf4j-version>1.7.21</org.slf4j-version>  </properties>  <dependencies>  <!-- Servlet -->  <dependency>  <groupId>javax.servlet</groupId>  <artifactId>javax.servlet-api</artifactId>  <version>3.1.0</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>javax.servlet.jsp</groupId>  <artifactId>javax.servlet.jsp-api</artifactId>  <version>2.2.1</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>javax.servlet</groupId>  <artifactId>jstl</artifactId>  <version>1.2</version>  </dependency>  <!-- Spring -->  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>${org.springframework-version}</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-webmvc</artifactId>  <version>${org.springframework-version}</version>  </dependency>  <!-- Logging -->  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>slf4j-api</artifactId>  <version>${org.slf4j-version}</version>  </dependency>  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>jcl-over-slf4j</artifactId>  <version>${org.slf4j-version}</version>  <scope>runtime</scope>  </dependency>  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>slf4j-log4j12</artifactId>  <scope>runtime</scope>  <version>${org.slf4j-version}</version>  </dependency>  <dependency>  <groupId>log4j</groupId>  <artifactId>log4j</artifactId>  <scope>runtime</scope>  <version>1.2.17</version>  </dependency>  <!-- Test -->  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <scope>test</scope>  <version>4.12</version>  </dependency>  </dependencies>  <build>  <plugins>  <plugin>  <artifactId>maven-compiler-plugin</artifactId>  <configuration>  <source>1.8</source>  <target>1.8</target>  </configuration>  </plugin>  </plugins>  </build>    </project> |

本講習における Spring Framework をはじめとした各ライブラリの構成は、

Spring IO 2.0.3-Release に準拠する。

＃Maven設定ファイル（pom.xml）を更新するとプロジェクトのリビルドが開始される。

＃ビルドの進捗は右下に地味に表示されているだけなので見落とさないこと

＃ビルドに数秒～数分がかかる場合があるが、終わるまで待っているほうがよい。

## Spring定義ファイル（その１） application-context-biz.xml を作成

１）Spring定義ダイル格納フォルダ作成

|  |
| --- |
| /src/main/webapp/WEB-INF/spring （新規作成） |

２）application-context-biz.xmlファイル新規作成 ＋ コーディング

|  |
| --- |
| /src/main/webapp/WEB-INF/spring/application-context-biz.xml （新規作成） |
| <?xml version="1.0" encoding="UTF-8"?>  <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:context="http://www.springframework.org/schema/context"  xsi:schemaLocation="  http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd">    <!-- Defines shared resources visible to all other web components -->  <context:component-scan base-package="jp.sample.bookmgr.biz" />    </beans> |

## Spring定義ファイル（その２） application-context-web.xml を作成

１）application-context-web.xmlファイル新規作成 ＋ コーディング

|  |
| --- |
| /src/main/webapp/WEB-INF/spring/application-context-web.xml （新規作成） |
| <?xml version="1.0" encoding="UTF-8"?>  <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:mvc="http://www.springframework.org/schema/mvc"  xmlns:context="http://www.springframework.org/schema/context"  xsi:schemaLocation="  http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/mvc  http://www.springframework.org/schema/mvc/spring-mvc.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd">  <!-- Defines this servlet's request-processing infrastructure -->  <context:component-scan base-package="jp.sample.bookmgr.web" />    <!-- Enables the Spring MVC @Controller programming model -->  <mvc:annotation-driven />  <!-- Resource mappings -->  <mvc:resources mapping="/resources/\*\*" location="/resources/" />  <!-- Resolves views selected for rendering by @Controllers to .jsp resources in the /WEB-INF/views directory -->  <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property name="prefix" value="/WEB-INF/views/" />  <property name="suffix" value=".jsp" />  </bean>  </beans> |

## Log4j 設定ファイル log4j.xml を作成

１）log4j.xmlファイル新規作成 ＋ コーディング

|  |
| --- |
| /src/main/resources/log4j.xml （新規作成） |
| <?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE log4j:configuration PUBLIC "-//APACHE//DTD LOG4J 1.2//EN" "log4j.dtd">  <log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">  <!-- Appenders -->  <appender name="console" class="org.apache.log4j.ConsoleAppender">  <param name="Target" value="System.out" />  <layout class="org.apache.log4j.PatternLayout">  <param name="ConversionPattern" value="%-5p: %c - %m%n" />  </layout>  </appender>    <!-- Application Loggers -->  <logger name="jp.sample.bookmgr.biz">  <level value="debug" />  </logger>  <logger name="jp.sample.bookmgr.web">  <level value="debug" />  </logger>  <!-- 3rdparty Loggers -->  <logger name="org.springframework.core">  <level value="info" />  </logger>    <logger name="org.springframework.beans">  <level value="info" />  </logger>    <logger name="org.springframework.context">  <level value="info" />  </logger>  <logger name="org.springframework.web">  <level value="info" />  </logger>  <!-- Root Logger -->  <root>  <priority value="warn" />  <appender-ref ref="console" />  </root>    </log4j:configuration> |

このタイミングで、画面下の「Markers」タブを参照して、エラーが検出されていたら、以下の操作を試してみること。

[Project Explorer] の [bookmgr] を右クリック → Maven → Update Project

数秒でプロジェクトが更新されエラーが消えるはず。

＃エラーが残る場合は声掛けしてください。

# Hello Spring アプリケーション作成



## Hello Spring 画面コントロールクラス作成

１）Hello Spring 画面コントロールクラス（HelloController.java）新規作成

|  |
| --- |
| /src/main/java/ で右クリック → New → Class  ● Java Class 画面  ---------------  Package: 「jp.sample.bookmgr.web.controller」  Name: 「HelloController」 と入力して Finish  --------------- |

２）Hello Spring 画面コントロールクラス（HelloController.java）のコーディング

|  |
| --- |
| /src/main/java/jp.sample.bookmgr.web.controller.HelloController.java （新規作成） |
| package jp.sample.bookmgr.web.controller;  import org.springframework.stereotype.Controller;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RequestMethod;  /\*\*  \* Hello Spring 画面コントローラクラス  \*  \* @author 長住@NTT-AT  \* @version 1.0  \*/  @Controller  public class HelloController {    /\*\*  \* Hello Spring 画面コントローラ  \*  \* @return 画面JSP名  \*/  @RequestMapping(value = "/", method = RequestMethod.GET)  public String helloSpring() throws Exception {  return "hello";  }  } |

## Hello Spring 画面View作成

１）Viewファイル格納フォルダ作成

|  |
| --- |
| /src/main/webapp/WEB-INF/views （新規作成） |

２）Hello Spring画面 View（hello.jsp）新規作成

|  |
| --- |
| bookmgr/src/main/webapp/WEB-INF/views で右クリック → New → JSP File  ● New JSP File 画面  ---------------  File Name: に「hello.jsp」 と入力して Finish  --------------- |

３）Hello Spring画面 View（hello.jsp）コーディング

|  |
| --- |
| /src/main/webapp/WEB-INF/views/hello.jsp （新規作成） |
| <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>Hello Spring 画面</title>  </head>  <body>  <h1>Hello Spring</h1>  きちんとみれましたでしょうか？  </body>  </html> |

## Hello World 画面表示テスト

|  |
| --- |
| ① プログラム起動方法（ここではEclipseにセットアップした Tomcat Server 上でサンプルプログラムを起動）  プロジェクト右クリック → Run as → Run on Server  ■Run On Server 画面  ---------------  「Tomcat 8.5 Server at localhost」 を選択 → [Next]  「bookmgr」 が Configerd に入っていることを確認して [Finish]  ---------------  **〈注意〉上記画面の Always use this server when running this project オプションは入れないほうが吉（後々）**  ② プログラム起動確認  Eclipseの内部ブラウザが立上り，以下のURLでHello worldが表示されれば完了．  アドレスは、http://localhost:8080/ bookmgr/ |

