

Loeng 3

Servlet API, Lombok,
projekti testid

Java Servlet (termin)

- Serveerib sisu (veebis)
- Spetsifitseeritud

Java SE

```
package hello;
```

```
public class Hello {
```

```
    public static void main(String[] args) {  
        System.out.println("Hello!");  
    }
```

```
}
```

```
> javac Hello.java
```

```
> java hello.Hello
```

Servlet

```
@WebServlet("/hello")
public class HelloServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request,
                          HttpServletResponse response)
        throws IOException {

        ...

    }
}
```

Väljund

```
@WebServlet("/hello")
public class HelloServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request,
                          HttpServletResponse response)
        throws IOException {

        response.getWriter().print("Hello!");
    }
}
```

Url

http://localhost:8080/hello

```
@WebServlet("/hello")
public class HelloServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request,
                          HttpServletResponse response)
        throws IOException {

        ...

    }
}
```

Url

http://localhost:8080/v1/api/orders

```
@WebServlet("/v1/api/orders")
public class HelloServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request,
                          HttpServletResponse response)
        throws IOException {

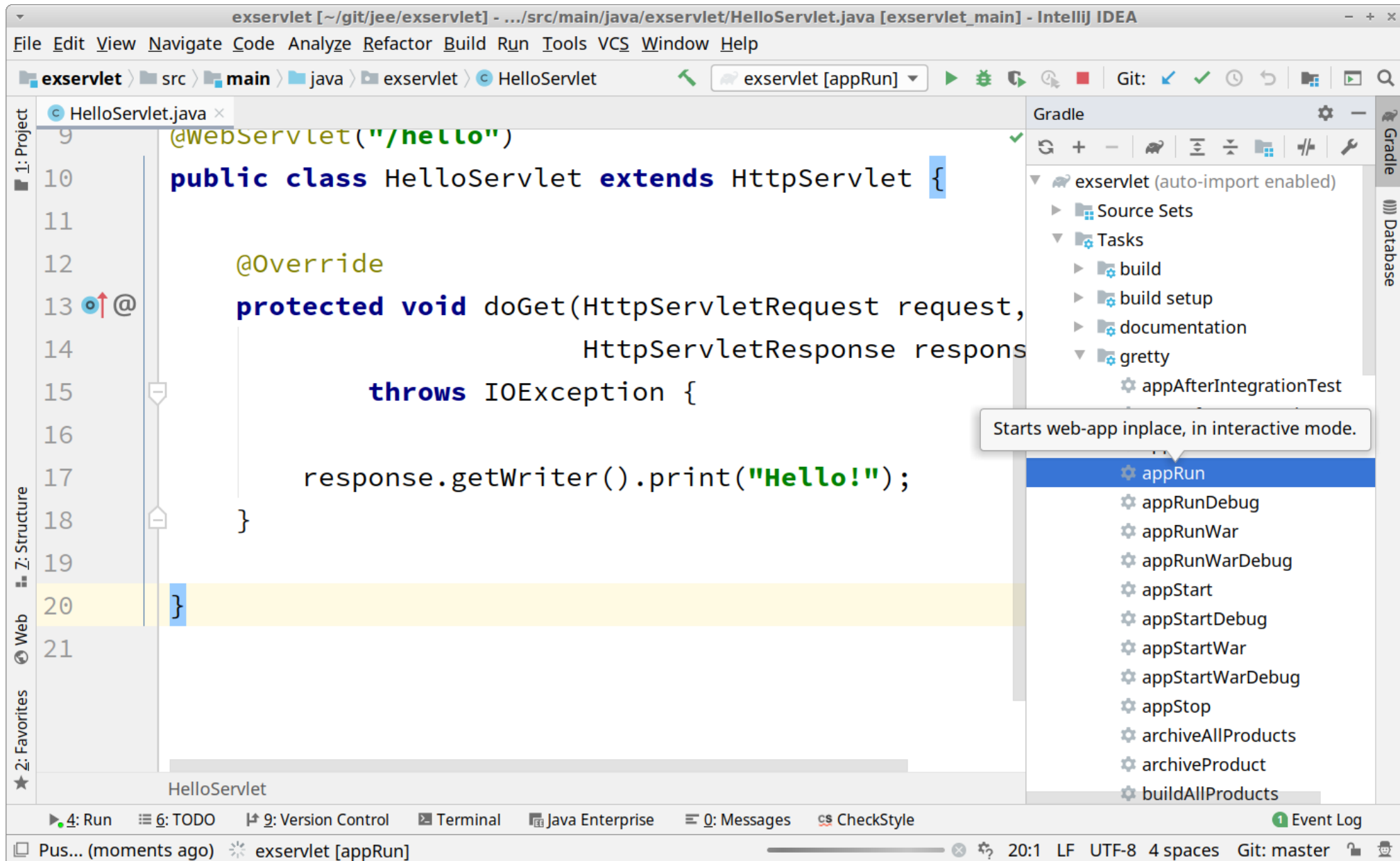
        ...

    }
}
```

Server

- Jetty, Tomcat, Oracle WebLogic, etc

Rakenduse käivitamine



Rakenduse käivitamine

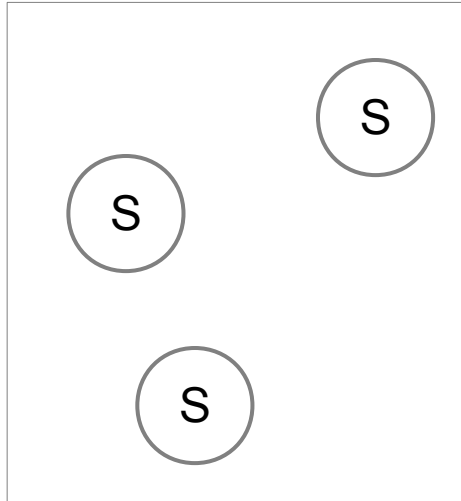
Gradle sakk paremas ääres -> Tasks -> gretty -> appRun

alternatiivne task appStart

Rakenduse laadimine

Server (Jetty)

Servlet Context 1



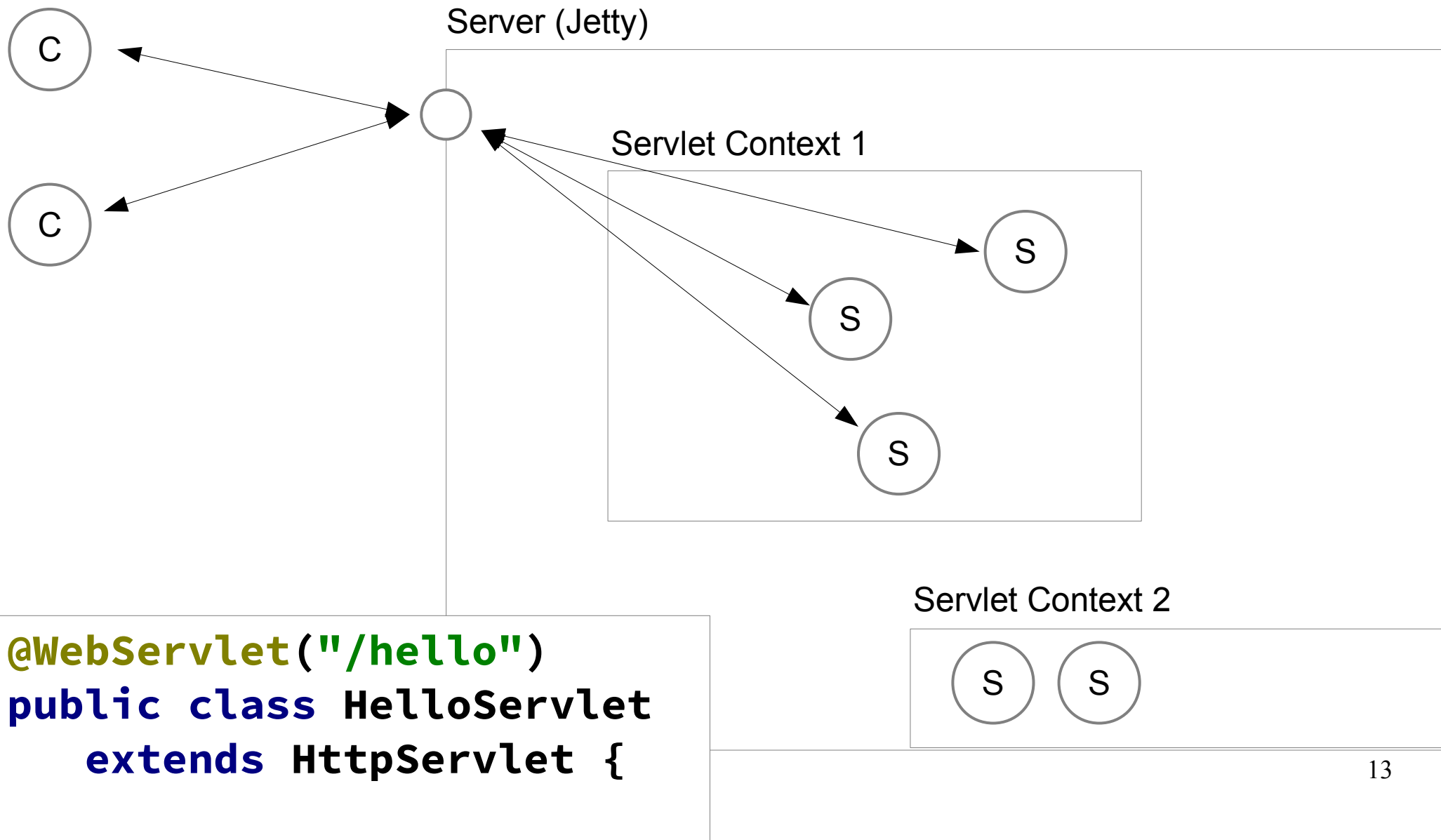
hello.war

```
`-- WEB-INF/
    |-- classes/
    |    `-- test
    |         |-- HelloServlet.class
    |         |-- OtherServlet.class
    |         |-- Util.class
    |         |-- ...
    `-- lib/
```

API-põhine arhitektuur

```
@WebServlet("/hello")  
public class Hello extends HttpServlet {  
  
    @Override protected void doGet(...  
  
    @Override protected void doPost(...  
  
    @Override protected void doPut(...  
  
    @Override protected void delete(...  
}
```

Servlet API



GET parameetrid

...hello?name=Alice&age=..

```
protected void doGet(HttpServletRequest request,  
                        HttpServletResponse response)  
    throws IOException {  
  
    String name = request.getParameter("name");  
}
```

Http protokoll

GET /api/orders/1 HTTP/1.1
Host: localhost:8080

HTTP/1.1 200 OK
Date: Sun, 23 Feb 2014 08:41:22
GMT

...

Content-Type: application/json

{ "id": 1, ... }

Projekti kirjeldusest

Päringu näide:

POST /api/orders HTTP/1.1

Host: localhost:8080

Content-Type: application/json

{ "orderNumber": "A123" }

Postman

- <https://chrome.google.com/webstore/detail/postman/fhbjggbiflinjbdgggehcdcdcbncdoddop?hl=en>

Andmevahetus

- Http protokoll on tekstipõhine
- Java rakenduses sooviksime töötada tüübitud ja objekti kujul andmetega

Json

- JavaScript Object Notation
- <https://en.wikipedia.org/wiki/JSON>

```
{  
  "id" : 1,  
  "title" : "Covert Json",  
  "hidden" : false,  
  "tags" : ["java", "json"],  
  "author": {  
    "firstName": "Jill",  
    "lastName": "Smith"  
  }  
}
```

Json (tekst) <-> Java

```
{  
  "id" : 1,  
  "title" : "Covert Json",  
  "hidden" : false,  
  "tags" : ["java", "json"],  
  "author": {  
    "firstName": "Jill",  
    "lastName": "Smith"  
  }  
}
```

```
public class Post {  
    private Long id;  
    private String title;  
    private boolean hidden;  
    private List<String> tags;  
    private Author author;
```

Accessor methods (getters and setters)

```
public class Post {  
    private Long id;  
    private String title;  
    private boolean hidden;  
    private List<String> tags;  
  
    public Long getId() {  
        return id;  
    }  
  
    public void setId(Long id) {  
        this.id = id;  
    }  
  
    ...  
}
```

ObjectMapper teek

build.gradle

...

```
dependencies {  
    implementation  
        'com.fasterxml.jackson.core:jackson-databind:2.11.2'
```

```
    ...
```

```
}
```

...

Java -> Json

```
Post post = new Post();  
post.setId(1L);  
post.setTitle("Covert Json");  
post.setHidden(false);  
post.setTags(Arrays.asList("java", "json"));  
  
String json = new ObjectMapper().writeValueAsString(post);
```

Json -> Java

```
String json = ...
```

```
Post post = new ObjectMapper()  
            .readValue(json, Post.class);
```


Json -> Java

```
String json = ...
```

```
Post post = new ObjectMapper()  
            .readValue(json, Post.class);
```

```
Customer c = new ObjectMapper()  
             .readValue(json, Customer.class);
```

```
Customer c = new ObjectMapper()  
            .readValue(json, Post.class);
```

Viga kompileerimisel!

Kõrgem abstraktsiooni tase (Spring MVC)

```
@PostMapping("/api/posts")  
public void save(@RequestBody Post post) {  
    // ...  
}
```

Kõrgem abstraktsiooni tase (Spring MVC)

DefaultHandlerExceptionResolver:140 Resolved
[org.springframework.web.HttpMediaTypeNotSupportedException:
Content type 'application/json;charset=UTF-8' not supported]

Kõrgem abstraktsiooni tase (Spring MVC)

DefaultHandlerExceptionResolver:140 Resolved
[org.springframework.http.converter.HttpMessageNotReadableException: **JSON parse error**: Unrecognized token 's': was expecting ('true', 'false' or 'null'); nested exception is com.fasterxml.jackson.core.JsonParseException: Unrecognized token 's': was expecting ('true', 'false' or 'null')
at [Source: (PushbackInputStream); line: 1, column: 11]]

Sisend ja väljund

...hello?name=Alice&age=..

```
protected void doGet(HttpServletRequest request, ...  
  
    String name = request.getParameter("name");  
  
    response.getWriter().print("...");  
  
    System.out.println("debug message");  
}
```

Sisend

...hello?name=Alice&age=..

```
protected void doPost(HttpServletRequest request, ...  
  
    // GET või POST vormi parameeter  
    String name = request.getParameter("name");  
  
    // päise parameeter  
    String host = request.getHeader("Host");  
  
    // kogu päringu sisu  
    InputStream is = request.getInputStream();  
}
```

Stream

- Pikkus pole teada
- Lugeda saab ühe korra

Stream stringiks

```
BufferedReader buffer =  
    new BufferedReader(new InputStreamReader(is));  
  
String input = buffer.lines()  
    .collect(Collectors.joining("\n"));
```

NB! Kaotab lõpust reavahetuse!

Streamist lugemine

```
protected void doPost(HttpServletRequest request, ...

    String body = request.getReader()
                        .lines()
                        .collect(Collectors.joining("\n"));

    ...

    ... = request.getReader()
            .lines()
            .collect(Collectors.joining("\n"));

    ...

}
```

NB! Problem!

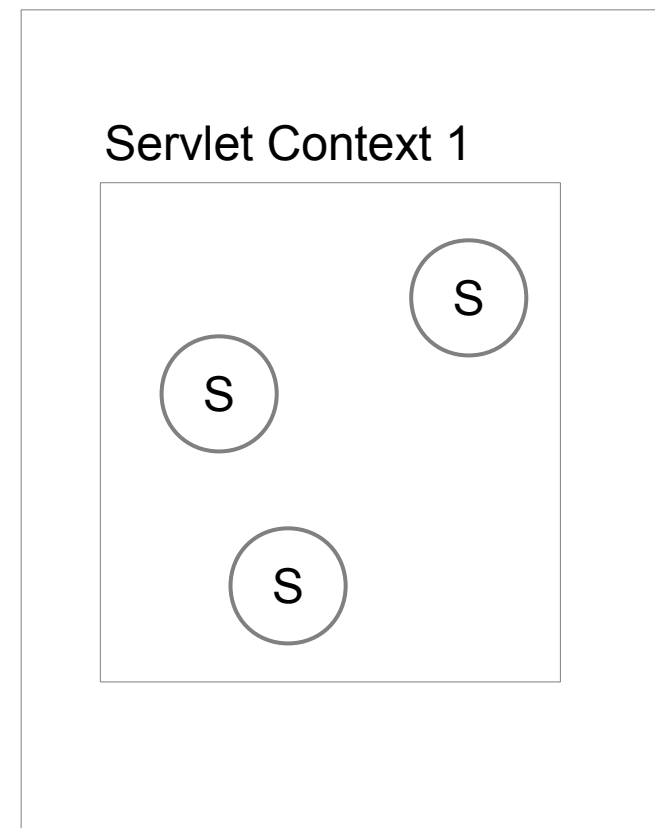
Kõrgem abstraktsiooni tase (Spring MVC)

```
@PostMapping("/api/posts")  
public void save(@RequestBody LargeObject data) {  
    // ...  
}
```

Servleti elutsükkel

- Igast servletist on rakenduse eluaja jooksul üks isend (instance)

Server (Jetty)



Servleti elutsükkel

```
public class HelloServlet extends HttpServlet {  
  
    private int reloadCount = 0;  
  
    protected void doGet(HttpServletRequest request, ...  
        reloadCount++;  
    }  
}
```

Servleti elutsükkel

```
public class HelloServlet extends HttpServlet {  
    private int userSpecificData = ...  
    protected void doGet(HttpServletRequest request, ...  
}
```

NB! Probleemne kood!

Servleti elutsükkel

- `init()`
- `doGet()`, `doPost()`, `doPut()`, `doDelete()`
- `destroy()`

Globaalsed tegevused

- `init()`
- ...

Listener-id

- ServletContextListener
- HttpSessionListener

ServletContextListener

@WebListener

public class MyListener **implements** ServletContextListener {

@Override

public void contextInitialized(ServletContextEvent sce) {
 System.out.println("Context initialized");
}

@Override

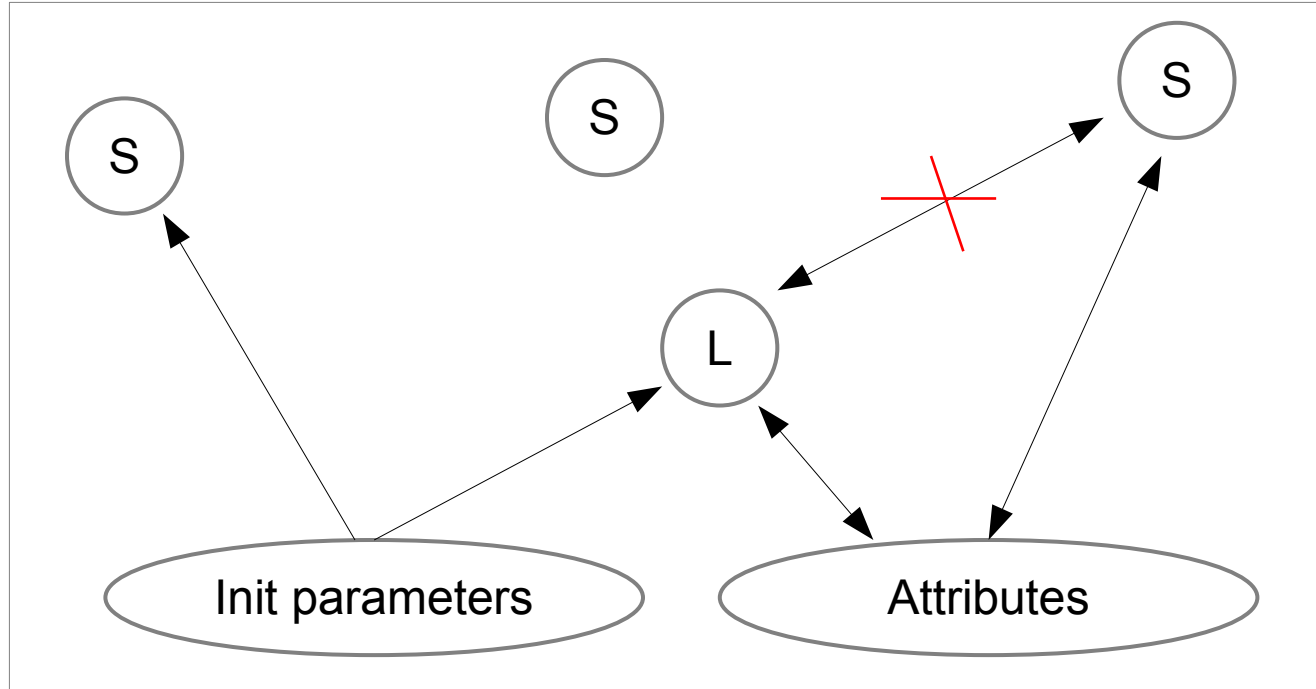
public void contextDestroyed(ServletContextEvent sce) {
 System.out.println("Context destroyed");
}

}

Servlet Context

Server

Servlet Context



Init Parameters (web.xml)

hello.war

```
`-- WEB-INF/  
    |-- classes/  
    |    |-- test  
    |    |-- HelloServlet.class  
    |-- lib/  
    |-- web.xml
```

ServletContext servlet-ist

@Override

```
protected void doGet(HttpServletRequest request,  
                        HttpServletResponse response)  
    throws IOException {  
  
    var context = getServletContext();
```

ServletContext listener-ist

```
@Override  
public void contextInitialized(ServletContextEvent sce) {  
    var context = sce.getServletContext();
```

ServletContext API

- `setAttribute("<name>", <value>);`
- `getAttribute("<name>");`
- `addServlet("<name>", <servlet instance>);`
- ...

Filter

```
@WebFilter("/*")
public class ProfilingFilter implements Filter {

    @Override
    public void doFilter(ServletRequest request,
                        ServletResponse response,
                        FilterChain chain)
        throws IOException {

        long millisBefore = System.currentTimeMillis();

        chain.doFilter(request, response);

        System.out.println(System.currentTimeMillis()
                           - millisBefore);
    }
}
```

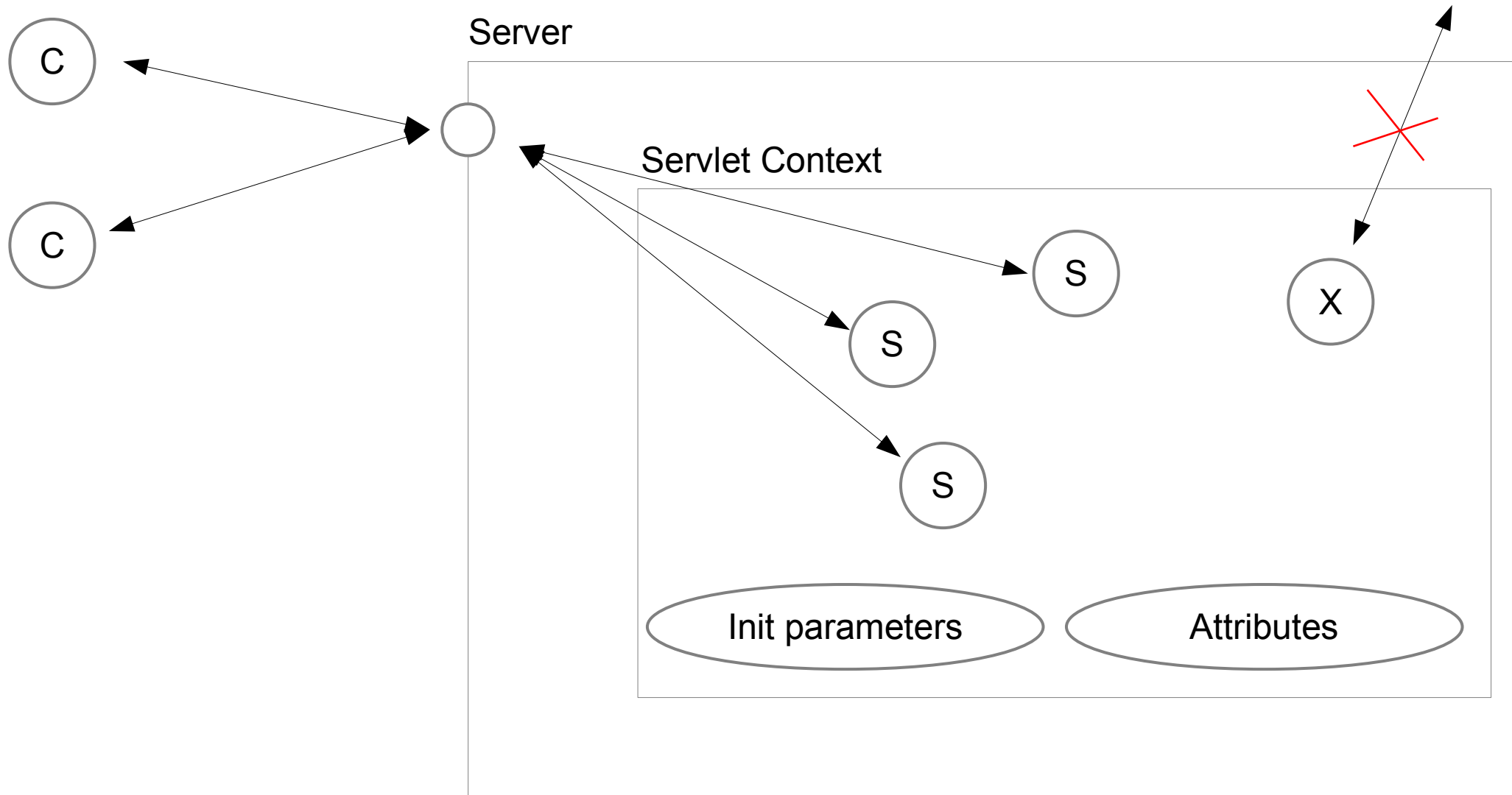
Kus filtreid kasutatakse

- Juurdepääsu kontroll
- Logimine
- Sisendi/väljundi transformeerimine
(nt. pakkimine)
- Jne.

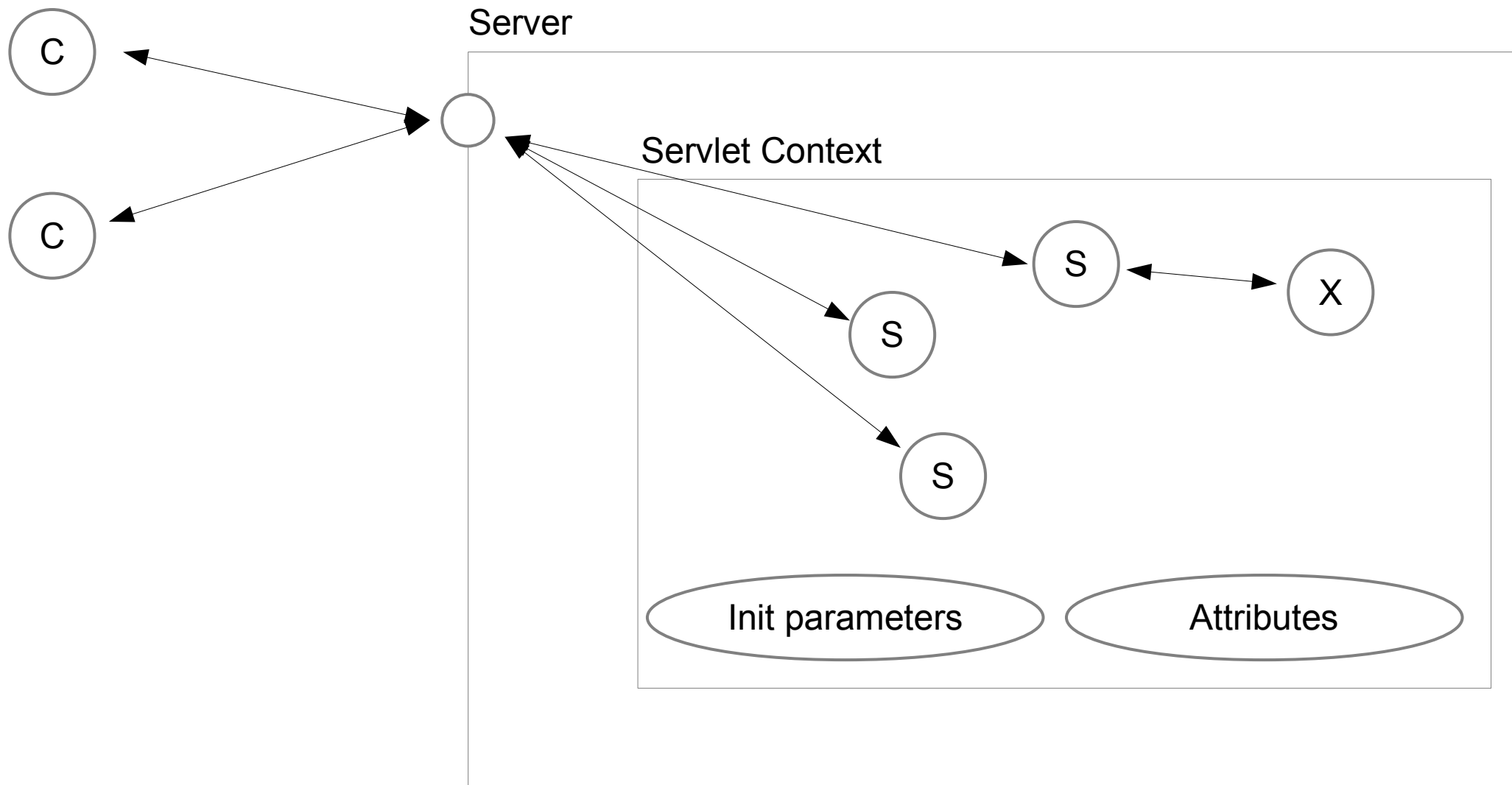
Java veebirakenduste API

- Servlet
- Listener
- Filter
- ServletContext

Java veebirakenduste API



Java veebirakenduste API



Lombok

@Getter

public class Customer {

private Long id;

@Setter

private String firstName;

@Setter

private String lastName;

private String type;

private String code;

private List<Phone> phones;

}

Lombok

@Data

```
public class Customer {  
    private Long id;  
    private String firstName;  
    private String lastName;  
    private String type;  
    private String code;  
  
    private List<Phone> phones;  
}
```

Lombok

```
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Customer {
    private Long id;
    private String firstName;
    private String lastName;
}
```

Annotation

```
// this class needs setters
public class Customer {
    private Long id;
    private String firstName;
    private String lastName;
}
```

Lombok

- IDEA - plugin, annotation processing, teek
- Gradle - teek

```
dependencies {
```

```
...
```

```
compileOnly 'org.projectlombok:lombok:1.18.12'
```

```
annotationProcessor 'org.projectlombok:lombok:1.18.12'
```


Projekti 3. osa

- Tähtaeg 20. september (22:00)
- Tag: hw03
- Peab vastama stiilireeglitele
- Peab läbima testid
- Kõik kood läheb samasse icd0011 reposse

Projekti 3. osa

Päringu näide:

POST /api/orders HTTP/1.1

Host: localhost:8080

Content-Type: application/json

```
{ "orderNumber": "A123" }
```

Projekti 3. osa

Vastuse näide:

Content-Type: application/json

```
{ "id": "1", "orderNumber": "A123" }
```

Projekti 3. osa

Json-ist info kätte saamine tuleb teha ilma väliste teekideta.

Projekti 3. osa

Päringu näide:

POST [/api/parser](#) HTTP/1.1

Host: localhost:8080

Content-Type: application/json

```
{ "gd": 2, "gdea": 2, "poi": "qs", "kdra": "qs" }
```

Vastuse näide:

HTTP/1.1 200 OK

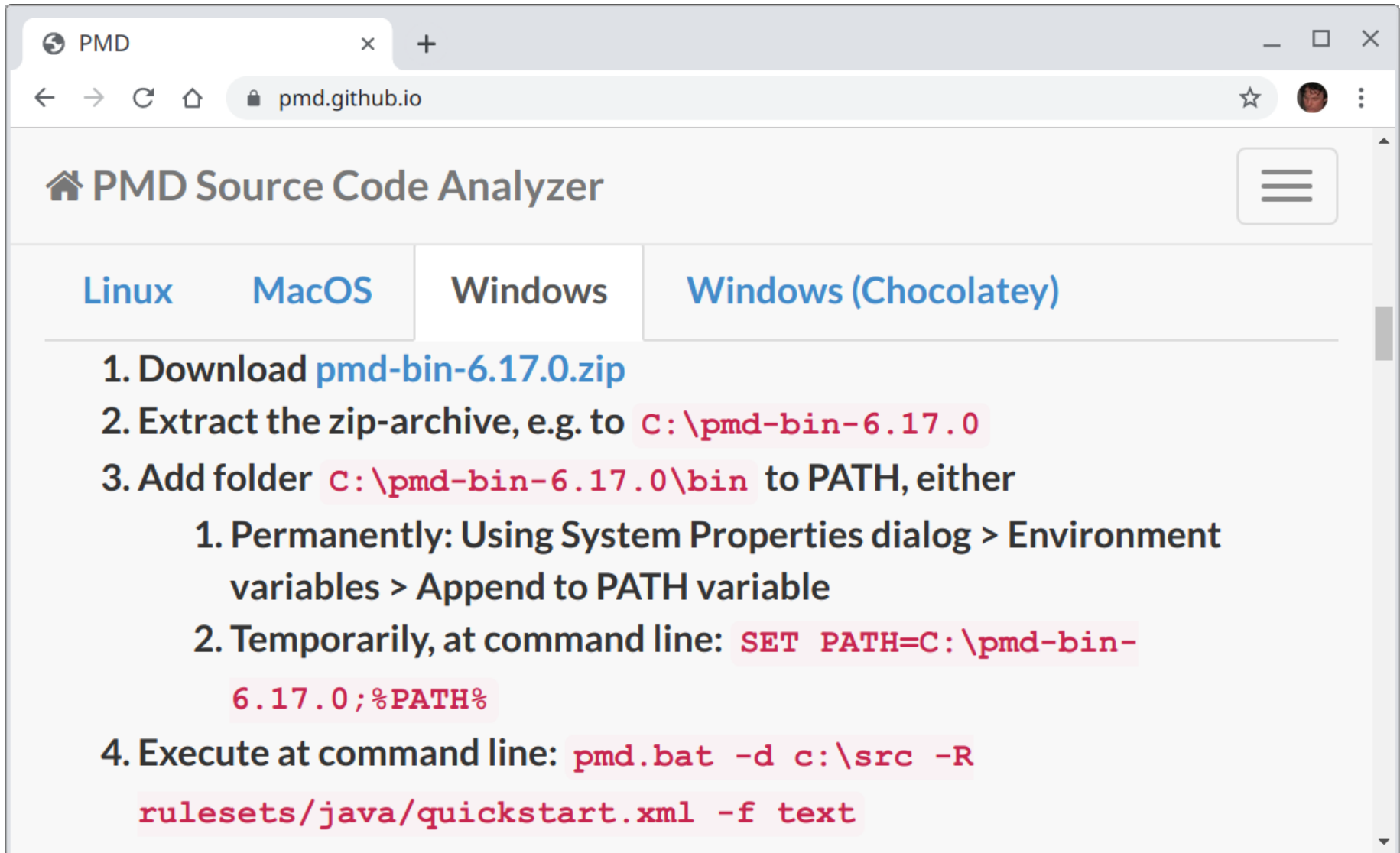
Content-Type: application/json

```
{ "gd": 2, "gdea": 4, "poi": "qs", "kdra": "kdra" }
```

Stiilireeglid

- PMD (<https://pmd.github.io/>)
- <https://bitbucket.org/mkalmo/hwtests/src/master/pmd/ruleset.xml>

Stiilireeglid



The screenshot shows a web browser window with the address bar displaying 'pmd.github.io'. The page title is 'PMD Source Code Analyzer'. Below the title, there are four tabs: 'Linux', 'MacOS', 'Windows', and 'Windows (Chocolatey)'. The 'Windows' tab is selected. The content area lists four steps for installation:

1. Download `pmd-bin-6.17.0.zip`
2. Extract the zip-archive, e.g. to `C:\pmd-bin-6.17.0`
3. Add folder `C:\pmd-bin-6.17.0\bin` to PATH, either
 1. Permanently: Using System Properties dialog > Environment variables > Append to PATH variable
 2. Temporarily, at command line: `SET PATH=C:\pmd-bin-6.17.0;%PATH%`
4. Execute at command line: `pmd.bat -d c:\src -R rulesets/java/quickstart.xml -f text`

Stiilireeglid

The screenshot shows the IntelliJ IDEA IDE interface. The main editor displays the `HelloServlet.java` file, which is part of the `exservlet` project. The code is as follows:

```
15      HttpServletResponse response)
16      throws IOException {
17
18      String first_name;
19
20      response.getWriter().print("Hello!");
21  }
22
```

The `String first_name;` line is highlighted in yellow. The `response.getWriter().print("Hello!");` line is highlighted in yellow. The `throws IOException {` line is highlighted in blue.

The PMD (Plugin for Microsoft Dynamics) window is open at the bottom, showing the results of a code analysis. It indicates 3 violations in 4 scanned files using 1 ruleset (3 violations). The violations are:

- UnusedImports. Avoid unused import statement. (4, 1) HelloServlet in exservlet
- UnusedLocalVariable. Detects when a local variable is declared but never used. (18, 16) HelloServlet.doGet() in exservlet
- LocalVariableNamingConventions. Configurable naming conventions for local variable declarations and other locally-scoped variables. This rule reports variable declarations which do not match the regex that applies to their specific kind (e.g. final variable, or catch-clause parameter). Each regex can be configured through properties. By default this rule uses the standard Java naming convention (Camel case). (18, 16) HelloServlet.doGet() in exservlet

The `LocalVariableNamingConventions` violation is highlighted in blue. A tooltip is visible over the `LocalVariableNamingConventions` violation, providing details about the rule and its configuration.

The status bar at the bottom shows the time as 20:46, the file encoding as UTF-8, the indentation as 4 spaces, and the current branch as Git: master.

Projekti testid

<https://bitbucket.org/mkalmo/hwtests>