FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

dbai

Semi-structured Data

11 - XSLT

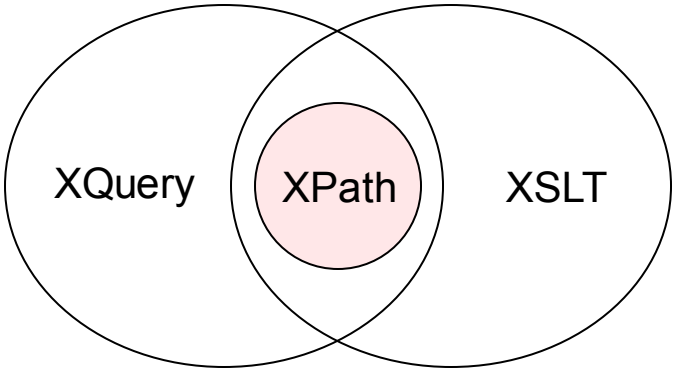
Andreas Pieris and Wolfgang Fischl, Summer Term 2016

Outline

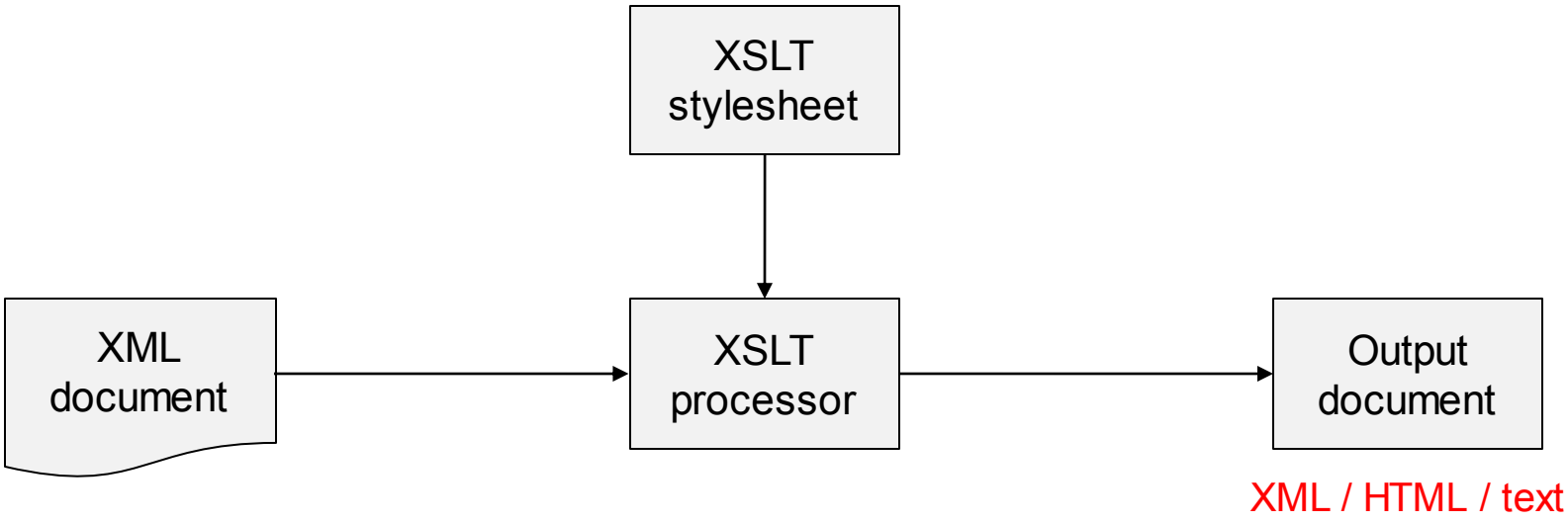
- What is XSLT?
- XSLT at First Glance
- XSLT Templates
- Creating Output
- Further Features

What is XSLT?

- XSL = eXtensible Stylesheet Language
- XSL = stylesheet language for XML (as CSS for HTML)
- XSLT = XSL Transformations
- XSLT is used to transform a source XML document into a target XML/HTML/text document
- XSLT uses XPath for navigation
- XSLT is a W3C standard



How XSLT Works?



- Define a transformation with an XSLT document (which is an XML document)
- Apply this transformation on an input document using an XSLT processor

XSLT at First Glance

```
<courses>
    <course semester="Summer">
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
    <course semester="Winter">
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
</courses>
```

```
<html>
  <head>
    <title>Lectures Overview</title>
  </head>
  <body>
    <h1>DBAI Lectures</h1>
    <table>
      <tr><th>Semester</th><th>Title</th>
          <th>Date / Time</th><th>Location</th></tr>
      <tr><td>Summer</td><td>SSD</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
      <tr><td>Winter</td><td>Databases</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
    </table>
  </body>
</html>
```

XSLT at First Glance

```
<courses>
    <course semester="Summer">
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
    <course semester="Winter">
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
</courses>
```

```
<html>
  <head>
    <title>Lectures Overview</title>
  </head>
  <body>
    <h1>DBAI Lectures</h1>
    <table>
      <tr><th>Semester</th><th>Title</th>
          <th>Date / Time</th><th>Location</th></tr>
      <tr><td>Summer</td><td>SSD</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
      <tr><td>Winter</td><td>Databases</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
    </table>
  </body>
</html>
```

XSLT at First Glance

```
<courses>
    <course semester="Summer">
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
    <course semester="Winter">
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
</courses>
```

```
<html>
  <head>
    <title>Lectures Overview</title>
  </head>
  <body>
    <h1>DBAI Lectures</h1>
    <table>
      <tr><th>Semester</th><th>Title</th>
          <th>Date / Time</th><th>Location</th></tr>
      <tr><td>Summer</td><td>SSD</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
      <tr><td>Winter</td><td>Databases</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
    </table>
  </body>
</html>
```

XSLT at First Glance

```
<courses>
    <course semester="Summer">
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
    <course semester="Winter">
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
        <location> HS8 </location>
    </course>
</courses>
```

```
<html>
  <head>
    <title>Lectures Overview</title>
  </head>
  <body>
    <h1>DBAI Lectures</h1>
    <table>
      <tr><th>Semester</th><th>Title</th>
          <th>Date / Time</th><th>Location</th></tr>
      <tr><td>Summer</td><td>SSD</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
      <tr><td>Winter</td><td>Databases</td>
          <td>Thursday, 09:15</td><td>HS8</td></tr>
    </table>
  </body>
</html>
```

XSLT at First Glance

Basic principle: templates match the input document, and define the output

```
<xsl:template  match="courses">
    <html>
        <head>
            <title>Lectures Overview</title></head>
        <body>
            <h1>DBAI Lectures</h1>
            <table>
                <tr><th>Semester</th><th>Title</th>
                    <th>Date / Time</th><th>Location</th></tr>
                <xsl:apply-templates  select="course"/>
            </table>
        </body>
    </html>
</xsl:template>
```

XSLT at First Glance

Basic principle: templates match the input document, and define the output

```
<xsl:template match="course">
  <tr><td><xsl:value-of select="@semester"/></td>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="day"/>, <xsl:value-of select="time"/></td>
      <td><xsl:value-of select="location"/></td>
  </tr>
</xsl:template>
```

XSLT Documents

```
<?xml version="1.0"?>
<xsl:stylesheet  version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output  version="html"/>
    <xsl:template  match="…">
        …
    </xsl:template>

    <xsl:template  match="…">
        …
    </xsl:template>
</xsl:stylesheet>
```

XSLT documents

are

XML documents

Up to Now

- What is XSLT?
- XSLT at First Glance
- XSLT Templates
- Creating Output
- Further Features

XSLT Templates

- A template matches an element node

  <xsl:template match="*">

- A template is applied if matches

- Does not match child nodes automatically

  <xsl:apply-templates select="childnode"/>

- Best practice
  - Define a template for the root node
  - Apply templates for child-elements starting from the root

Template Matching

If a template matches an element

- Template is executed
- By default, no templates for the subtree is called, except when explicitly applied (<xsl:apply-templates>)

```
<xsl:template  match="person">
        Hello!!!
</xsl:template>
```

```
<person>
        <name> Andreas Pieris </name>
        <email> pieris@dbai.tuwien.ac.at </email>
</person>
```

Hello!!!

Apply Templates

<xsl:apply-templates> applies templates for child elements

```
<xsl:template match="person">
        Hello!!!
        <xsl:apply-templates select="name"/>
</xsl:template>

<xsl:template match="name">
        <xsl:value-of select="."/>
</xsl:template>
```

```
<person>
        <name> Andreas Pieris </name>
        <email> pieris@dbai.tuwien.ac.at </email>
</person>
```

Hello!!!

Andreas Pieris

Apply Templates

<xsl:apply-templates> applies templates for child elements

```
<xsl:template match="person">
        Hello!!!
        <xsl:apply-templates select="*"/>
</xsl:template>

<xsl:template match="name">
        <xsl:value-of select="."/>
</xsl:template>
```

```
<person>
        <name> Andreas Pieris </name>
        <email> pieris@dbai.tuwien.ac.at </email>
</person>
```

Hello!!!

Andreas Pieris

pieris@dbai.tuwien.ac.at

Default Templates

- XSLT defines default templates that are always present

- Default templates are as follows
  - For root and elements: apply templates for child elements
  - For text elements: copy content to the output
  - For attributes: copy value to the output

- To override the behaviour of a default template create a template for an element

Default Templates

<xsl:apply-templates> executes templates for child elements

```
<xsl:template match="person">
        Hello!!!
        <xsl:apply-templates select="*"/>
</xsl:template>

<xsl:template match="name">
        <xsl:value-of select="."/>
</xsl:template>
```

```
<person>
        <name> Andreas Pieris </name>
        <email> pieris@dbai.tuwien.ac.at </email>
</person>
```

Hello!!!

Andreas Pieris

pieris@dbai.tuwien.ac.at

Priorities

- Exactly one template is executed
- In case of more than one templates, a priority value decides which template is executed
- The XPath expression in the match attribute indicates the priority
- More specific XPath expressions have higher priority

Priorities

```
<xsl:template match="person">
        Hello!!!
        <xsl:apply-templates select="*"/>
</xsl:template>

<xsl:template match="name">
        <xsl:value-of select="."/>
</xsl:template>

<xsl:template match="*">
        Conflict!!!
</xsl:template>
```

Hello!!!

Andreas Pieris

Conflict!!!

```
<person>
        <name> Andreas Pieris </name>
        <email> pieris@dbai.tuwien.ac.at </email>
</person>
```

Priorities

```
<xsl:template match="person">
        Hello!!!
        <xsl:apply-templates select="*"/>
</xsl:template>

<xsl:template match="name">
        <xsl:value-of select="."/>
</xsl:template>

<xsl:template match="name">
        Conflict!!!
</xsl:template>
```

Ambiguous rule match

```
<person>
        <name> Andreas Pieris </name>
        <email> pieris@dbai.tuwien.ac.at </email>
</person>
```

Any Problem?

```
<xsl:template match="person">
        <xsl:apply-templates select="name"/>
        <xsl:apply-templates select="email"/>
</xsl:template>

<xsl:template match="name">
        Here it goes this
        <xsl:value-of select="."/>
</xsl:template>

<xsl:template match="email">
        Here it goes this
        <xsl:value-of select="."/>
</xsl:template>
```

same content

Named Templates

```
<xsl:template  match="person">
        <xsl:apply-templates  select="name"/>
        <xsl:apply-templates  select="email"/>
</xsl:template>

<xsl:template  match="name | email">
        <xsl:call-template  name="output"/>
</xsl:template>

<xsl:template  name="output">
        Here it goes this
        <xsl:value-of select="."/>
</xsl:template>
```

Outline

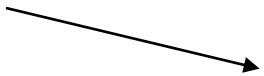
- What is XSLT?
- XSLT at First Glance
- XSLT Templates
- Creating Output
- Further Features

Creating Output

- xsl:element
- xsl:attribute
- xsl:text
- xsl:comment

Creating Output - xsl:element

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```



```
<courses>
    <SSD>
        <day> Thursday </day>
        <time> 09:15 </time>
    </SSD>
    <Databases>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </Databases>
</courses>
```

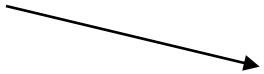
Creating Output - xsl:element

```
<xsl:template match="courses">
        <courses>
            <xsl:apply-templates select="course"/>
        </courses>
</xsl:template>

<xsl:template match="course">
    <xsl:element name="{title/text()}">
            <day> <xsl:value-of select="day"/> </day>
            <time> <xsl:value-of select="time"/> </time>
        </xsl:element>
</xsl:template>
```

Creating Output - xsl:attribute

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```



```
<courses>
    <SSD day="Thursday">
        <time> 09:15 </time>
    </SSD>
    <Databases day="Tuesday">
        <time> 09:15 </time>
    </Databases>
</courses>
```

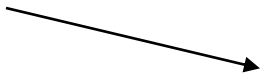
Creating Output - xsl:attribute

```
<xsl:template match="courses">
    <courses>
        <xsl:apply-templates select="course"/>
    </courses>
</xsl:template>

<xsl:template match="course">
    <xsl:element name="{title/text()}">
        <xsl:attribute name="day">
            <xsl:value-of select="day"/>
        </xsl:attribute>
        <time> <xsl:value-of select="time"/> </time>
    </xsl:element>
</xsl:template>
```

Creating Output - xsl:text

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```



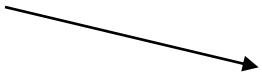
```
<courses>
    <SSD  day="Thursday">
        Starts at
        09:15
    </SSD>
    <Databases day="Tuesday">
        Starts at
        09:15
    </Databases>
</courses>
```

Creating Output - xsl:text

```
<xsl:template match="courses">
    <courses>
        <xsl:apply-templates select="course"/>
    </courses>
</xsl:template>
<xsl:template match="course">
    <xsl:element name="{title/text()}">
        <xsl:attribute name="day">
            <xsl:value-of select="day"/>
        </xsl:attribute>
        <xsl:text>
            Starts at
            <xsl:value-of select="time"/>
        </xsl:text>
    </xsl:element>
</xsl:template>
```

Creating Output - xsl:comment

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```



```
<courses>
    <!– Course Description -->
    <SSD day="Thu">
        Starts at 09:15
    </SSD>
    <Databases day="Tue">
        Starts at 09:15
    </Databases>
</courses>
```

Creating Output - xsl:comment

```
<xsl:template match="courses">
    <courses>
        <xsl:comment> Course Description </xsl:comment>
        <xsl:apply-templates select="course"/>
    </courses>
</xsl:template>

<xsl:template match="course">
    <xsl:element name="{title/text()}">
        <xsl:attribute name="day">
            <xsl:value-of select="day"/>
        </xsl:attribute>
        <xsl:text> Starts at <xsl:value-of select="time"/> </xsl:text>
    </xsl:element>
</xsl:template>
```

Outline

- What is XSLT?
- XSLT at First Glance
- XSLT Templates
- Creating Output
- Further Features

Further Features

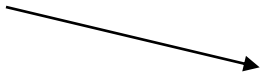
- xsl:for-each
- xsl:sort
- xsl:if
- xsl:choose
- xsl:variable

Further Features - xsl:for-each

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```

The structure of the course element is not known

We only know that the first child of each course element is title



```
<courses>
    <SSD>
        <day> Thursday </day>
        <time> 09:15 </time>
    </SSD>
    <Databases>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </Databases>
</courses>
```

Further Features - xsl:for-each

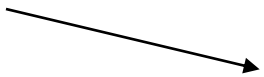
```
<xsl:template  match="courses">
        <courses>
              <xsl:apply-templates  select="course"/>
        </courses>
</xsl:template>

<xsl:template  match="course">
        <xsl:element  name="{title/text()}">
              <xsl:for-each select="*[position() &gt; 1]">
                    <xsl:copy-of select="."/>
              </xsl:for-each>
        </xsl:element>
</xsl:template>
```

creates a copy of the selected node

Further Features - xsl:sort

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```



Sort by name

```
<courses>
    <Databases>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </Databases>
    <SSD>
        <day> Thursday </day>
        <time> 09:15 </time>
    </SSD>
</courses>
```

Further Features - xsl:sort

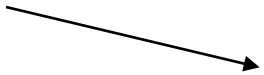
```
<xsl:template match="courses">
    <courses>
        <xsl:for-each select="course">
            <xsl:sort select="title" order="ascending"/>
            <xsl:element name="{title/text()}">
                <xsl:for-each select="*[position() &gt; 1]">
                    <xsl:copy-of select="."/>
                </xsl:for-each>
            </xsl:element>
        </xsl:for-each>
    </courses>
</xsl:template>
```

Further Features - xsl:if

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```

The structure of the course element is not known

The first child of each course element is not necessarily title



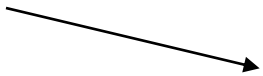
```
<courses>
    <Databases>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </Databases>
    <SSD>
        <day> Thursday </day>
        <time> 09:15 </time>
    </SSD>
</courses>
```

Further Features - xsl:if

```
<xsl:template match="courses">
    <courses>
        <xsl:for-each select="course">
            <xsl:sort select="title" order="ascending"/>
            <xsl:element name="{title/text()}">
                <xsl:for-each select="*">
                    <xsl:if test="local-name() != 'title' ">
                        <xsl:copy-of select="."/>
                    </xsl:if>
                </xsl:for-each>
            </xsl:element>
        </xsl:for-each>
    </courses>
</xsl:template>
```

Further Features - xsl:choose

```
<courses>
    <course>
        <title> SSD </title>
        <day> Thursday </day>
        <time> 09:15 </time>
    </course>
    <course>
        <title> Databases </title>
        <day> Tuesday </day>
        <time> 09:15 </time>
    </course>
</courses>
```



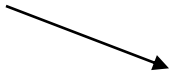
```
<courses>
    <Databases>
        Tuesday, 09:15.
    </Databases>
    <SSD>
        Thursday, 09:15.
    </SSD>
</courses>
```

Further Features - xsl:choose

```
<xsl:template match="courses">
    <courses>
        <xsl:for-each select="course">
            <xsl:sort select="title" order="ascending"/>
            <xsl:element name="{title/text()}">
                <xsl:for-each select="*">
                    <xsl:if test="local-name() != 'title' ">
                        <xsl:value-of select="."/>
                        <xsl:choose>
                            <xsl:when test="position() = last()">.</xsl:when>
                            <xsl:otherwise>,</xsl:otherwise>
                        </xsl:choose>
                    </xsl:if>
                </xsl:for-each>
            </xsl:element>
        </xsl:for-each>
    </courses>
</xsl:template>
```

Further Features - xsl:variable

```
<university>
    <courses>
        <course taughtBy="L1">
            <title> SSD </title>
            <day> Thursday </day>
            <time> 09:15 </time>
        </course>
        <course taughtBy="L2">
            <title> Databases </title>
            <day> Tuesday </day>
            <time> 09:15 </time>
        </course>
    </courses>
    <lecturers>
        <lecturer id="L1">Andreas Pieris</lecturer>
        <lecturer id="L2">Reinhard Pichler</lecturer>
    </lecturers>
</university>
```



```
<courses>
    SSD taught by Andreas Pieris
    Databases taught by Reinhard Pichler
</courses>
```

Further Features - xsl:variable

```
<xsl:template match="university">
    <xsl:apply-templates select="courses"/>
</xsl:template>

<xsl:template match="courses">
    <courses>
        <xsl:apply-templates select="course"/>
    </courses>
</xsl:template>

<xsl:template match="course">
    <xsl:value-of select="title"/> taught by <xsl:value-of select="//lecturer[@id = @taughtBy]"/>
</xsl:template>
```

Result:

```
<courses>
    SSD taught by
    Databases taught by
</courses>
```

?

Further Features - xsl:variable

```
<xsl:template match="university">
        <xsl:apply-templates select="courses"/>
</xsl:template>

<xsl:template match="courses">
        <courses>
                <xsl:apply-templates select="course"/>
        </courses>
</xsl:template>

<xsl:template match="course">
        <xsl:variable name="var" select="@taughtBy"/>
        <xsl:value-of select="title"/> taught by <xsl:value-of select="//lecturer[@id = $var]"/>
</xsl:template>
```

Sum Up

- What is XSLT?
- XSLT at First Glance
- XSLT Templates
- Creating Output
- Further Features

FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

dbai

Semi-structured Data

Examination

Andreas Pieris and Wolfgang Fischl, Summer Term 2016

Structure of the Exam

- Schema validation - multiple choice (12 points)
- General multiple choice questions (15 points)
- Exercise (48 points)
    - XML document given
    - Create schema
    - XPath expressions
    - XQuery expressions
    - SAX/DOM
    - XSLT

Grading

- Everybody registered in TISS can attend the exam

- For a positive certificate
    - Total points (exam + exercises) >= 50 (out of 100)
    - Exam points >= 37.5

- Exercise points are valid only for this semester (the next 4 exams)

Grading

Total 100 points

Points (p)	Grade
p < 50 or < 37.5 on exam	5
50 <= p <= 61	4
61 < p <= 74	3
74 < p <= 87	2
p > 87	1

Grading

- A certificate is issue for very exam attempt
- If an exercise was uploaded to TUWEL, then a (negative) certificate will be issued even without an exam attempt

Exam Dates

- Main exam date: Wednesday, 22nd of June, 18:30 – 20:30

- Three further dates next semester:
  - 28/10/2016, 15:00 – 17:00
  - 28/11/2016, 15:00 – 17:00
  - 10/01/2017, 18:00 – 20:00

- **Always check in TISS for the dates and times!!!**

Registration / Deregistration

- Registration/Deregistration is available in TISS
- Registration starts ~2 weeks before the exam and ends 2 days before the exam
- Deregistration is possible until the day before the exam
- Lecture Halls:
  - For the first exam we have two lecture halls (FH HS 1 & HS 8)
  - Register for the main exam date (Haupttermin)
  - People will be distributed on the day before the exam
  - Distribution is announced in TISS and please check in which room you have to go

After the Exam

- Announcement of results
  - Example solutions will be uploaded to the web
  - Results are published in TUWEL
  - Approximate time to correct the exams is 4 weeks

- Exam inspection
  - After the results are published or after the holiday (for the exams in January and June)
  - Location and date/time will be announced via TISS