合肥学院
HEFEI UNIVERSITY

Distributed Computing

Homework 2: HTTP Proxy Servlet

Thomas Weise · 汤卫思

tweise@hfuu.edu.cn · http://www.it-weise.de

Hefei University, South Campus 2
Faculty of Computer Science and Technology
Institute of Applied Optimization
230601 Shushan District, Hefei, Anhui, China
Econ. & Tech. Devel. Zone, Jinxiu Dadao 99

合肥学院 南艳湖校区/南2区
计算机科学与技术系
应用优化研究所
中国 安徽省 合肥市 蜀山区 230601
经济技术开发区 锦绣大道99号

Outline

1. Introduction
2. HTTP Proxies
3. Task

website

Overview

- Learn about HTTP [1] Proxies
- Learn how to use Maven
- Learn about URLs and HTML [2, 3]
- Better understand the interaction of web browsers and web servers
- Learn how to build Stand-Alone Java Servlets [4]

Introduction

- The web is boring.

Introduction

- The web is boring.
- It is time to “enhance” the WWW!!

Introduction

- The web is boring.
- It is time to “enhance” the WWW!!
    - How often did you search for the wrong terms in Baidu?

Introduction

- The web is boring.
- It is time to “enhance” the WWW!!
    - How often did you search for the wrong terms in Baidu?
    - How often did you visit the wrong website?

Introduction

- The web is boring.
- It is time to “enhance” the WWW!!
    - How often did you search for the wrong terms in Baidu?
    - How often did you visit the wrong website?
    - How often did you find that the text of a website displayed in your browser could be improved?

Introduction

- The web is boring.
- It is time to “enhance” the WWW!!
    - How often did you search for the wrong terms in Baidu?
    - How often did you visit the wrong website?
    - How often did you find that the text of a website displayed in your browser could be improved?
- Behold: All of these problems can be solved with a HTTP Proxy Server!

HTTP Proxy Server

- What is a HTTP Proxy Server?

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
    - When you open a web page, normally your web browser sends a HTTP request to a web server.

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
    - When you open a web page, normally your web browser sends a HTTP request to a web server.
    - In this request, it asks for the web resource (page) it wants to display.

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
    - When you open a web page, normally your web browser sends a HTTP request to a web server.
    - In this request, it asks for the web resource (page) it wants to display.
    - The web server then sends back a HTTP response, which contains the resource (web page) as body.

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
  - When you open a web page, normally your web browser sends a HTTP request to a web server.
  - In this request, it asks for the web resource (page) it wants to display.
  - The web server then sends back a HTTP response, which contains the resource (web page) as body.
- What is different with a HTTP Proxy?

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
    - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
  - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
  - The proxy then sends a HTTP request to the web server.

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
    - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
    - The proxy then sends a HTTP request to the web server.
    - The web server sends the resource (page) back to the proxy as body of its HTTP response.

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
  - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
  - The proxy then sends a HTTP request to the web server.
  - The web server sends the resource (page) back to the proxy as body of its HTTP response.
  - The proxy sends the resource back to the web browser as body of its HTTP response.

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
  - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
  - The proxy then sends a HTTP request to the web server.
  - The web server sends the resource (page) back to the proxy as body of its HTTP response.
  - The proxy sends the resource back to the web browser as body of its HTTP response.
- What is this good for?

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
    - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
    - The proxy then sends a HTTP request to the web server.
    - The web server sends the resource (page) back to the proxy as body of its HTTP response.
    - The proxy sends the resource back to the web browser as body of its HTTP response.
- What is this good for?
    - caching

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
    - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
    - The proxy then sends a HTTP request to the web server.
    - The web server sends the resource (page) back to the proxy as body of its HTTP response.
    - The proxy sends the resource back to the web browser as body of its HTTP response.
- What is this good for?
    - caching
    - protecting anonymity of users

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
    - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
    - The proxy then sends a HTTP request to the web server.
    - The web server sends the resource (page) back to the proxy as body of its HTTP response.
    - The proxy sends the resource back to the web browser as body of its HTTP response.
- What is this good for?
    - caching
    - protecting anonymity of users
    - monitoring, limiting access to internet, firewall

HTTP Proxy Server

- What is a HTTP Proxy Server?
- Well, what is your browser normally doing?
- What is different with a HTTP Proxy?
  - When you open a web page, normally your web browser sends the HTTP request to the HTTP proxy instead of the web server.
  - The proxy then sends a HTTP request to the web server.
  - The web server sends the resource (page) back to the proxy as body of its HTTP response.
  - The proxy sends the resource back to the web browser as body of its HTTP response.
- What is this good for?
  - caching
  - protecting anonymity of users
  - monitoring, limiting access to internet, firewall
  - ...and, of course, for "enhancing" the WWW!

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
  - (re)construct the original URL that the client originally wanted

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
  - (re)construct the original URL that the client originally wanted, including
  - the “parameters” of the URL, if any, i.e., the query string

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
  - (re)construct the original URL that the client originally wanted, including
  - the "parameters" of the URL, if any, i.e., the query string and
  - read the HTTP header fields and adapt them to what our proxy can support

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
    - (re)construct the original URL that the client originally wanted, including
    - the "parameters" of the URL, if any, i.e., the query string and
    - read the HTTP header fields and adapt them to what our proxy can support (e.g., maybe we cannot support persistent connections)

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
  - (re)construct the original URL that the client originally wanted, including
  - the "parameters" of the URL, if any, i.e., the query string and
  - read the HTTP header fields and adapt them to what our proxy can support (e.g., maybe we cannot support persistent connections)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL and
  - send the query from the original URL

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL and
  - send the query from the original URL and
  - forward the HTTP header fields from the original request coming from the browser

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL and
  - send the query from the original URL and
  - forward the HTTP header fields from the original request coming from the browser (which is necessary to include, for instance, cookies needed to enable sessions).

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL and
    - send the query from the original URL and
    - forward the HTTP header fields from the original request coming from the browser (which is necessary to include, for instance, cookies needed to enable sessions).
- From the connection **B** read the answer

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer, including
    - the HTTP status code

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer, including
  - the HTTP status code,
  - the HTTP header fields

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer, including
    - the HTTP status code,
    - the HTTP header fields, and
    - the message body

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer, including
  - the HTTP status code,
  - the HTTP header fields, and
  - the message body, i.e., the actual resource requested by the browser in step 1

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer, including
  - the HTTP status code,
  - the HTTP header fields, and
  - the message body, i.e., the actual resource requested by the browser in step 1, e.g., a HTML page or an image

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer, including
  - the HTTP status code,
  - the HTTP header fields, and
  - the message body, i.e., the actual resource requested by the browser in step 1, e.g., a HTML page or an image
- Then it will forward these elements read from connection **B** via connection **A** back to the client

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer
- Then it will forward these elements read from connection **B** via connection **A** back to the client, i.e.,
    - the HTTP status code

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer
- Then it will forward these elements read from connection **B** via connection **A** back to the client, i.e.,
    - the HTTP status code,
    - the HTTP header fields

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer
- Then it will forward these elements read from connection **B** via connection **A** back to the client, i.e.,
    - the HTTP status code,
    - the HTTP header fields (this includes, e.g., cookies sent from the server to web browser to establish a session)

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer
- Then it will forward these elements read from connection **B** via connection **A** back to the client, i.e.,
  - the HTTP status code,
  - the HTTP header fields (this includes, e.g., cookies sent from the server to web browser to establish a session), and

How does a HTTP Proxy work?

- For each incoming HTTP connection **A** (from a client such as a web browser)
- If the request cannot be satisfied directly, open a new HTTP connection **B** to the host indicated by the original URL
- From the connection **B** read the answer
- Then it will forward these elements read from connection **B** via connection **A** back to the client, i.e.,
  - the HTTP status code,
  - the HTTP header fields (this includes, e.g., cookies sent from the server to web browser to establish a session), and
  - the actual message body.

How can we build this?

- Connection **A**

How can we build this?

- Connection **A**
  - Java Servlets allow us to accept an incoming HTTP request and send back a HTTP response

How can we build this?

- Connection **A**
  - Java Servlets allow us to accept an incoming HTTP request and send back a HTTP response
- Connection **B**

How can we build this?

- Connection **A**
- Connection **B**
  - The Java object `URL` allows us to instantiate a `URLConnection`

How can we build this?

- Connection **A**
- Connection **B**
  - The Java object `URL` allows us to instantiate a `URLConnection`
  - `URLConnection`s send a HTTP request to the destination host of a `URL` and receive the HTTP response

How can we build this?

- Connection **A**
- Connection **B**
  - The Java object `URL` allows us to instantiate a `URLConnection`
  - `URLConnection`s send a HTTP request to the destination host of a `URL` and receive the HTTP response
  - We can set HTTP header fields for the request and read the HTTP header fields and status code from the response

How can we build this?

- Connection **A**
- Connection **B**
  - The Java object `URL` allows us to instantiate a `URLConnection`
  - `URLConnection`s send a HTTP request to the destination host of a `URL` and receive the HTTP response
  - We can set HTTP header fields for the request and read the HTTP header fields and status code from the response
- In other words: We can implement a HTTP proxy as Java Servlet!

How can we build this?

- Connection **A**
- Connection **B**
- In other words: We can implement a HTTP proxy as Java Servlet!
- Deployment?

How can we build this?

- Connection **A**
- Connection **B**
- In other words: We can implement a HTTP proxy as Java Servlet!
- Deployment?
    - Normally, Java Servlets are deployed into a servlet container

How can we build this?

- Connection **A**
- Connection **B**
- In other words: We can implement a HTTP proxy as Java Servlet!
- Deployment?
  - Normally, Java Servlets are deployed into a servlet container
  - The Jetty [5] can be "embedded", i.e., packaged into a single `jar` archive *together* with the Java Servlets

How can we build this?

- Connection **A**
- Connection **B**
- In other words: We can implement a HTTP proxy as Java Servlet!
- Deployment?
  - Normally, Java Servlets are deployed into a servlet container
  - The Jetty [5] can be "embedded", i.e., packaged into a single `jar` archive *together* with the Java Servlets
  - With Maven, we can build a "fat `jar`" also including all dependencies and required other libraries.

How can we build this?

- Connection **A**
- Connection **B**
- In other words: We can implement a HTTP proxy as Java Servlet!
- Deployment?
    - Normally, Java Servlets are deployed into a servlet container
    - The Jetty[5] can be "embedded", i.e., packaged into a single `jar` archive *together* with the Java Servlets
    - With Maven, we can build a "fat `jar`" also including all dependencies and required other libraries.
    - This can then be executed stand-alone, like a normal `jar`

Implementation

- Don't worry. I have already done all of this for you.

Implementation

- Don't worry. I have already done all of this for you.
- It is part of my example code repository on GitHub

Implementation

- Don't worry. I have already done all of this for you.
- It is part of my example code repository on GitHub
- In folder `javaServlets/proxy` you can find an Eclipse project with Maven `pom`

Implementation

- Don't worry. I have already done all of this for you.
- It is part of my example code repository on GitHub
- In folder `javaServlets/proxy` you can find an Eclipse project with Maven `pom`
- At `https://github.com/thomasWeise/distributedComputingExamples/tree/master/javaServlets/proxy`, you find a detailed documentation and a usage guide

Implementation

- Don't worry. I have already done all of this for you.
- It is part of my example code repository on GitHub
- In folder `javaServlets/proxy` you can find an Eclipse project with Maven `pom`
- At `https://github.com/thomasWeise/distributedComputingExamples/tree/master/javaServlets/proxy`, you find a detailed documentation and a usage guide
- So what is the task?

Goal

- Modify the proxy server to “enhance” the web!

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser, e.g.,
    - by replacing some strings in the HTML body (but make sure to not damage tags or links. . .)

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser, e.g.,
    - by replacing some strings in the HTML body (but make sure to not damage tags or links. . .)
    - sometimes printing all text between certain tags backwards (but make sure to not damage tags or links. . .)

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
    - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser, e.g.,
        - by replacing some strings in the HTML body (but make sure to not damage tags or links. . .)
        - sometimes printing all text between certain tags backwards (but make sure to not damage tags or links. . .)
        - replace the names of some people with other words, such as 土豆

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser, e.g.,
    - by replacing some strings in the HTML body (but make sure to not damage tags or links...)
    - sometimes printing all text between certain tags backwards (but make sure to not damage tags or links...)
    - replace the names of some people with other words, such as 土豆
    - replace all `OK` buttons in all `<form>`s with `type="reset"` buttons...

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
    - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser, e.g.,
        - by replacing some strings in the HTML body (but make sure to not damage tags or links. . .)
        - sometimes printing all text between certain tags backwards (but make sure to not damage tags or links. . .)
        - replace the names of some people with other words, such as 土豆
        - replace all `OK` buttons in all `<form>`s with `type="reset"` buttons. . .
        - . . .

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser, e.g.,
    - by replacing some strings in the HTML body (but make sure to not damage tags or links. . .)
    - sometimes printing all text between certain tags backwards (but make sure to not damage tags or links. . .)
    - replace the names of some people with other words, such as 土豆
    - replace all `OK` buttons in all `<form>`s with `type="reset"` buttons. . .
    - . . .
  - instead of using the original query provided by the user, we may actually perform a different query

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
    - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser
    - instead of using the original query provided by the user, we may actually perform a different query, e.g.,
        - randomly add or remove search terms if the query is for Baidu or Bing

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser
  - instead of using the original query provided by the user, we may actually perform a different query, e.g.,
    - randomly add or remove search terms if the query is for Baidu or Bing
    - re-route queries, e.g., randomly switch between Bing and Baidu (requires proper translation of the respective URL syntaxes)

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser
  - instead of using the original query provided by the user, we may actually perform a different query, e.g.,
    - randomly add or remove search terms if the query is for Baidu or Bing
    - re-route queries, e.g., randomly switch between Bing and Baidu (requires proper translation of the respective URL syntaxes)
    - re-route queries, e.g., go to Tudou when Youku was queried, then switch to yet another platform

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser
  - instead of using the original query provided by the user, we may actually perform a different query, e.g.,
    - randomly add or remove search terms if the query is for Baidu or Bing
    - re-route queries, e.g., randomly switch between Bing and Baidu (requires proper translation of the respective URL syntaxes)
    - re-route queries, e.g., go to Tudou when Youku was queried, then switch to yet another platform
    - remember all requested URLs and sometimes (randomly) replace a requested URL with one from memory, i.e., take the browser back to an earlier visited page

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser
  - instead of using the original query provided by the user, we may actually perform a different query, e.g.,
    - randomly add or remove search terms if the query is for Baidu or Bing
    - re-route queries, e.g., randomly switch between Bing and Baidu (requires proper translation of the respective URL syntaxes)
    - re-route queries, e.g., go to Tudou when Youku was queried, then switch to yet another platform
    - remember all requested URLs and sometimes (randomly) replace a requested URL with one from memory, i.e., take the browser back to an earlier visited page
    - every now and then re-route to a very interesting fixed website, e.g., one showing the weather in Nicaragua.

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
    - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser
    - instead of using the original query provided by the user, we may actually perform a different query, e.g.,
        - randomly add or remove search terms if the query is for Baidu or Bing
        - re-route queries, e.g., randomly switch between Bing and Baidu (requires proper translation of the respective URL syntaxes)
        - re-route queries, e.g., go to Tudou when Youku was queried, then switch to yet another platform
        - remember all requested URLs and sometimes (randomly) replace a requested URL with one from memory, i.e., take the browser back to an earlier visited page
        - every now and then re-route to a very interesting fixed website, e.g., one showing the weather in Nicaragua.
        - ...

Goal

- Modify the proxy server to “enhance” the web!
- There are many ways in which we can assist the user with “enhancements”:
  - once the proxy receives the answer from the actual web server, if the answer is a web page (and not a image), we may “enhance” it before sending it on to the web browser
  - instead of using the original query provided by the user, we may actually perform a different query, e.g.,
    - randomly add or remove search terms if the query is for Baidu or Bing
    - re-route queries, e.g., randomly switch between Bing and Baidu (requires proper translation of the respective URL syntaxes)
    - re-route queries, e.g., go to Tudou when Youku was queried, then switch to yet another platform
    - remember all requested URLs and sometimes (randomly) replace a requested URL with one from memory, i.e., take the browser back to an earlier visited page
    - every now and then re-route to a very interesting fixed website, e.g., one showing the weather in Nicaragua.
    - ...
- As you can see, we can be of much assistance

Task

- Modify the Java Servlet Proxy example to implement at least one of the above “enhancements” or invent your “enhancement”.

Task

- Modify the Java Servlet Proxy example to implement at least one of the above “enhancements” or invent your “enhancement”.
- This enhancement must not break the web pages, i.e., cannot destroy links or the page structure

Task

- Modify the Java Servlet Proxy example to implement at least one of the above “enhancements” or invent your “enhancement”.
- This enhancement must not break the web pages, i.e., cannot destroy links or the page structure
- It must also not lead to any other form of error

Task

- Modify the Java Servlet Proxy example to implement at least one of the above “enhancements” or invent your “enhancement”.
- This enhancement must not break the web pages, i.e., cannot destroy links or the page structure
- It must also not lead to any other form of error
- Build a stand-alone `jar` with your implemented proxy servlet with Maven

Task

- Modify the Java Servlet Proxy example to implement at least one of the above "enhancements" or invent your "enhancement".
- This enhancement must not break the web pages, i.e., cannot destroy links or the page structure
- It must also not lead to any other form of error
- Build a stand-alone `jar` with your implemented proxy servlet with Maven
- Run your serverlet locally, set up the web browser to use your proxy, make screenshots (I demand 5 screenshots!)

Task

- Modify the Java Servlet Proxy example to implement at least one of the above "enhancements" or invent your "enhancement".
- This enhancement must not break the web pages, i.e., cannot destroy links or the page structure
- It must also not lead to any other form of error
- Build a stand-alone `jar` with your implemented proxy servlet with Maven
- Run your serverlet locally, set up the web browser to use your proxy, make screenshots (I demand 5 screenshots!)
- Submit your code, `jar`, and screenshots as a zip archive named `hw02_[your_student_id].zip` (where `[your_student_id]` is replaced with your student id) to me

Tips

- The `https://github.com/thomasWeise/distributedComputingExamples/tree/master/javaServlets/proxy` contains detailed instructions on how to build the `jar`, how to setup the web browser for using the proxy, and how to run the `jar` in the background without creating any visible window

Tips

- The `https://github.com/thomasWeise/distributedComputingExamples/tree/master/javaServlets/proxy` contains detailed instructions on how to build the `jar`, how to setup the web browser for using the proxy, and how to run the `jar` in the background without creating any visible window
- There are only two classes in the project, you need to modify `ProxyServlet`

Tips

- The `https://github.com/thomasWeise/distributedComputingExamples/tree/master/javaServlets/proxy` contains detailed instructions on how to build the `jar`, how to setup the web browser for using the proxy, and how to run the `jar` in the background without creating any visible window
- There are only two classes in the project, you need to modify `ProxyServlet`
- This class is extensively documented.

谢谢
Thank you

Thomas Weise [汤卫思]
tweise@hfuu.edu.cn
http://www.it-weise.de

Hefei University, South Campus 2
Institute of Applied Optimization
Shushan District, Hefei, Anhui, China

Caspar David Friedrich, “Der Wanderer über dem Nebelmeer”, 1818
http://en.wikipedia.org/wiki/Wanderer_above_the_Sea_of_Fog

Bibliography

Bibliography I

1. R. Fielding, J. Gettys, Jeffrey Mogul, H. Frystyk, L. Masinter, P. Leach, and Timothy John Berners-Lee. *Hypertext Transfer Protocol – HTTP/1.1*, volume 2616 of *Request for Comments (RFC)*. Network Working Group, June 1999. URL `http://tools.ietf.org/html/rfc2616`.
2. Dave Raggett, Arnaud Le Hors, and Ceriel J. H. Jacobs. *HTML 4.01 Specification*. W3C Recommendation. MIT/CSAIL (USA), ERCIM (France), Keio University (Japan): World Wide Web Consortium (W3C), December 24, 1999. URL `http://www.w3.org/TR/1999/REC-html401-19991224`.
3. Murray Altheim and Shane McCarron. *XHTML™ 1.1 – Module-based XHTML – Second Edition*. W3C Recommendation. MIT/CSAIL (USA), ERCIM (France), Keio University (Japan): World Wide Web Consortium (W3C), November 23, 2010. URL `http://www.w3.org/TR/2010/REC-xhtml11-20101123`.
4. Rajiv Mordani. *JSR 315: Java™ Servlet 3.0 Specification Version 3.0 Rev a (Maintenance Release)*, volume 315 of *Java Specification Requests (JSR)*. Maintenance release edition, December 2010. URL `http://download.oracle.com/otndocs/jcp/servlet-3.0-mrel-eval-oth-JSpec`.
5. *Jetty WebServer*. Fortitude Valley BC, QLD, Australia: codehause foundation and Riverview, NSW, Australia: Mort Bay Consulting Pty. Ltd, 1995. URL `http://jetty.codehaus.org/jetty/`.