Web Services
Overview

- Quick walkthrough of Web application architectures
  - WS-* Web Services
  - Representational State Transfer (REST)

- Exercise 2
  - Overview
  - Tasks
  - Hints & Anchors
Web Services

- Definition:

“A Web service is an application component accessible over open protocols”

Invoke calls

Offer Services

Client

Server
Web Services in a Nutshell

- **Lookup-Service**
  - **UDDI**

- **Client**
  - lookup
  - request/reply

- **Server**
  - publish

- **SOAP**
  - **WSDL**
For the exercise, we let the service publish its WSDL without going through a UDDI...
Web Services - WSDL Overview

- **WSDL:** [Web Services Description Language](https://en.wikipedia.org/wiki/Web_Services_Description_Language) describes:
  - What a Web service can do
  - Where it resides
  - How to invoke it

- Explore WSDL

Types, Messages, PortType, Binding, Service, Port, Definition
Programming WS-* Clients

- Most IDEs provide code generators

  **Server-side**
  - Java annotations
  - Automatic generation of WSDL file

  **Client-side**
  - Parsing of WSDL file
  - Automatic generation of client stubs
REST: **Representational State Transfer**

- REST is a lightweight architectural style for designing networked applications
  - HTTP 1.1 implements the REST architectural style
  - It uses HTTP for CRUD (Create/Read/Update/Delete) operations

- Platform independent
- Language independent
- Open standard-based

[http://geekandpoke.typepad.com/]
REST Architecture

- **Resources**: Identified by logical URIs
  - State and functionality are represented using resources
    
e.g., a sensor node: [http://vslab.inf.ethz.ch:8081/sunspots/Spot1](http://vslab.inf.ethz.ch:8081/sunspots/Spot1)

- **A web of resources**: Resources are linked
  - Similar to the interconnection of Web pages in the WWW
  - When relevant, resources should link to additional information
    - Resources should be kept simple

- **Stateless** communication protocol:
  - Each new request must carry all the information required to complete it
Assignment 2 – Overview

- **Objectives:**
  - Learn to develop distributed Web applications
  - Use the two different paradigms seen in the lecture:
    - Representational State Transfer (REST)
    - Web Services (WS-*)

- **Dates:**
  - Exercise begins: **Now** (October 6, 2014)
  - Exercise is due: **9:00 am, October 20, 2014**
Assignment 2 – System Setup

- Access Sun SPOTs through WS-* and REST
- Sun SPOTs: Wireless sensor nodes (temp, acc, light,...)
Assignment 2 – Task 1

Experimenting with RESTful Web Services (2P)

- Create an HTTP request
  - a) “manually” (i.e., without the use of an HTTP library)
  - b) using `org.apache.http.*`
- Use HTTP content negotiation to get machine-readable data
- Connect to a Sun SPOT and retrieve the temperature value
- **Hint:** Use AsyncTask to do network operations (be careful with accessing UI Elements!)
- **Hint:** Use the HTTP header “Connection: close” to avoid blocking behavior
Assignment 2 – Task 2

Experimenting with WS-* Web Services (2P)

- Explore WSDL, create SOAP requests
- Connect to a Sun SPOT and retrieve the temperature value.
- **Hint:** Apply hints from Task 1
- **Hint:** Use the Android version of the kSOAP2 library
- **Hint:** Important classes are: SoapObject, SoapSerializationEnvelope
- **Hint:** You do not have to implement the decoding of the WSDL file
Code Skeleton

- Interfaces for Sensors
  - Separate UI from logic
  - Increase of code reuse
  - Each subtask is a new class that implements the Sensor interface
Assignment 2 – Task 3

Network Traffic Analysis (1P)

- Learn how to use tools for network traffic analysis (e.g. Wireshark)
- Debugging tool for distributed software
Assignment 2 – Task 4, 5

Your Phone as a Server (3P)

- Implement a Web server on your phone that allows to access the sensors and actuators of the phone
- **Hint:** Use a Service to implement the server
- **Hint:** Use Intents and BroadcastReceiver, or Bound Services, to communicate between Service and Activity
- **Hint:** When you are using an existing WiFi network, make sure the ports you are using are not blocked!

Task 5: Mini-Test (2P)
Deliverables

- See exercise sheet for details
  - code.zip
  - answers.zip
  - wireshark.zip
Assignment 2 Hints - Relevant Terminology

- **Internet Media Types**
  - text/html, text/xml
  - application/xml, application/json

- **ROA – Resource-Oriented Architecture**
- **REST – Representational State Transfer**

- **SOA – Service-oriented Architecture**
- **SOAP – Simple Object Access Protocol**
- **WSDL – Web Services Description Language**
REST Hints

- http://www.infoq.com/articles/rest-introduction

- RESTful Web Services (Leonard Richardson und Sam Ruby)
  - Available at D-INFK library

- Apache HTTP library (simplest sample code alive... 😊)
Noteworthy Tools

- Firefox extensions
  - HttpRequester
  - Poster
  - RESTClient
  - SOA Client

- Chrome extensions
  - Simple REST client
Noteworthy Tools

- Android Debug Bridge (adb tool)
  - You can find the adb tool in `<sdk>/platform-tools/

- Android Emulator

- Setting up a port forwarding
  - `adb forward tcp:port1 tcp:port2`
  - forwards the local port port1 on the machine to port2 on the emulator.
  - Example: `adb forward tcp:12345 tcp:8088`