Engineering Internet Applications

Konstantinos Karasavvas

CITY College

October 15, 2014

Table of contents

1. Module Introduction
2. World Wide Web
3. Web Services
4. Web Applications

Module Introduction

- Background
  - about me
  - about you

Module Introduction

- Background
  - about me
  - about you
- Aims/Objectives
  - Understand basic principles of Internet Applications
  - Understand the architectural aspects
  - Describe in detail web development (using Ruby and Sinatra)
  - Understand the development process and organisation

Module Introduction

- Background
  - about me
  - about you
- Aims/Objectives
  - Understand basic principles of Internet Applications
  - Understand the architectural aspects
  - Describe in detail web development (using Ruby and Sinatra)
  - Understand the development process and organisation
- 3 hours per week
  - 2 hours lectures
  - 1 hour practical sessions (laptops/labs)

Module Introduction

- Background
  - about me
  - about you
- Aims/Objectives
  - Understand basic principles of Internet Applications
  - Understand the architectural aspects
  - Describe in detail web development (using Ruby and Sinatra)
  - Understand the development process and organisation
- 3 hours per week
  - 2 hours lectures
  - 1 hour practical sessions (laptops/labs)
- Assessment
  - Group project in Ruby/Sinatra (70%)
  - Quiz in Java (30%)

Module Introduction, cont.

- Resources
  - Why's (Poignant Guide) to Ruby
    - http://mislav.uniqpath.com/poignant-guide/
  - Mr. Neighborly's Humble Little Ruby Book
    - http://humblelittlerubybook.com/
  - Sinatra: Up and Running (http://it-ebooks.info/book/547/)
  - https://www.ruby-lang.org/en/
  - http://www.sinatrarb.com/
  - http://sinatra-book.gittr.com/
  - ...

Module Introduction, cont.

- Resources
  - Why's (Poignant Guide) to Ruby
    - http://mislav.uniqpath.com/poignant-guide/
  - Mr. Neighborly's Humble Little Ruby Book
    - http://humblelittlerubybook.com/
  - Sinatra: Up and Running (http://it-ebooks.info/book/547/)
  - https://www.ruby-lang.org/en/
  - http://www.sinatrarb.com/
  - http://sinatra-book.gittr.com/
  - ...
- Contact
  - kkarasavvas@gmail.com
  - Skype: kkarasavvas
  - Twitter: kkarasavvas
  - LinkedIn: Konstantinos Karasavvas

Module Introduction, cont.

- Resources
  - Why's (Poignant Guide) to Ruby
    - http://mislav.uniqpath.com/poignant-guide/
  - Mr. Neighborly's Humble Little Ruby Book
    - http://humblelittlerubybook.com/
  - Sinatra: Up and Running (http://it-ebooks.info/book/547/)
  - https://www.ruby-lang.org/en/
  - http://www.sinatrarb.com/
  - http://sinatra-book.gittr.com/
  - ...
- Contact
  - kkarasavvas@gmail.com
  - Skype: kkarasavvas
  - Twitter: kkarasavvas
  - LinkedIn: Konstantinos Karasavvas
- Ask, ask, and when in doubt ask!!!

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows
  - Basic knowledge of HTML (CSS, JS)

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows
  - Basic knowledge of HTML (CSS, JS)
  - Basic understanding of database design

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows
  - Basic knowledge of HTML (CSS, JS)
  - Basic understanding of database design
  - Source Code Management tool (preferably Git)

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows
  - Basic knowledge of HTML (CSS, JS)
  - Basic understanding of database design
  - Source Code Management tool (preferably Git)
  - Quick to learn new tools and technologies

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows
  - Basic knowledge of HTML (CSS, JS)
  - Basic understanding of database design
  - Source Code Management tool (preferably Git)
  - Quick to learn new tools and technologies
- For the future
  - Start a personal project

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows
  - Basic knowledge of HTML (CSS, JS)
  - Basic understanding of database design
  - Source Code Management tool (preferably Git)
  - Quick to learn new tools and technologies
- For the future
  - Start a personal project
  - Be passionate

Expected Skills

- For the module
  - Basic knowledge of desired OS
    - Unix-based (recommended)
    - Windows
  - Basic knowledge of HTML (CSS, JS)
  - Basic understanding of database design
  - Source Code Management tool (preferably Git)
  - Quick to learn new tools and technologies
- For the future
  - Start a personal project
  - Be passionate (work hard!)

Course Outline

- Week 1: Introduction to EIA and Ruby
- Week 2: Introduction to Ruby
- Week 3: Sinatra, MVC, (++) and Tiny App demonstrating Web Dev.
- Week 4: Environments, Sessions, Error handling, (++)
- Week 5: Associations, Migrations, Web Authentication, Web Services
- Week 6: Consolidation Week
- Week 7: Client-side / Presentation
- Week 8: Unit Testing, Functional Testing (++)
- Week 9: Web Development with Java
- Week 10: Web Development with Java
- Week 11: Web Development with Java
- Week 12: Revision Week

Coursework

- One project, in three phases / courseworks
  - requirement analysis and design (group)
  - basic implementation (group)
  - advanced implementation (individual)

Coursework

- One project, in three phases / courseworks
  - requirement analysis and design (group)
  - basic implementation (group)
  - advanced implementation (individual)
- Hard deadlines — no delays!
- Feedback is essential for following phases

Basics (Technical)

- Hypertext Transfer Protocol (HTTP)
  - request-response protocol
    - application layer (TCP/IP & OSI)
  - client-server model
  - `GET`, `POST`, `DELETE`, ...

```
1 $ telnet www.example.com 80
```

Basics (Technical)

- Hypertext Transfer Protocol (HTTP)
  - request-response protocol
    - application layer (TCP/IP & OSI)
  - client-server model
  - GET, POST, DELETE, ...

```
1 $ telnet www.example.com 80
2 Trying 192.0.43.10...
3 Connected to www.example.com.
4 Escape character is '^]'.
```

Basics (Technical)

- Hypertext Transfer Protocol (HTTP)
  - request-response protocol
    - application layer (TCP/IP & OSI)
  - client-server model
  - GET, POST, DELETE, ...

```
1 $ telnet www.example.com 80
2 Trying 192.0.43.10...
3 Connected to www.example.com.
4 Escape character is '^]'.
5 GET / HTTP/1.1
```

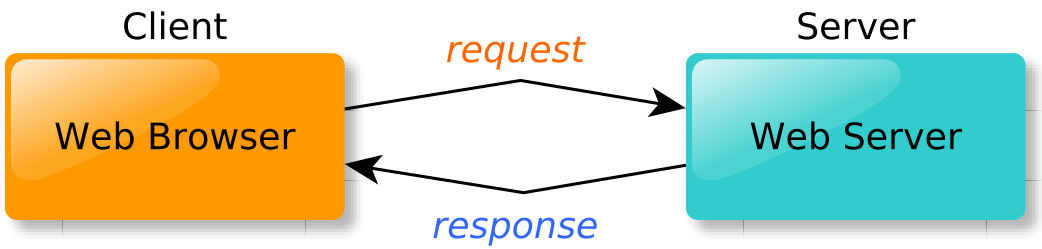
Basics (Technical)

- Hypertext Transfer Protocol (HTTP)
  - request-response protocol
    - application layer (TCP/IP & OSI)
  - client-server model
  - GET, POST, DELETE, ...

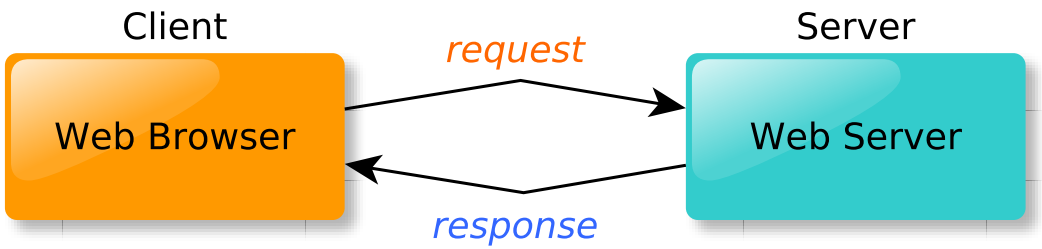
```
$ telnet www.example.com 80
Trying 192.0.43.10...
Connected to www.example.com.
Escape character is '^]'.
GET / HTTP/1.1

HTTP/1.0 302 Found
Location: http://example.iana.org
Server: BigIP
Connection: Keep-Alive
Content-Length: 0
```

Basics (Technical), cont.

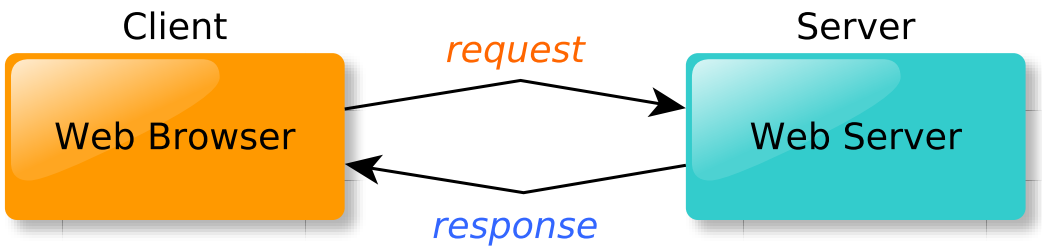


Basics (Technical), cont.



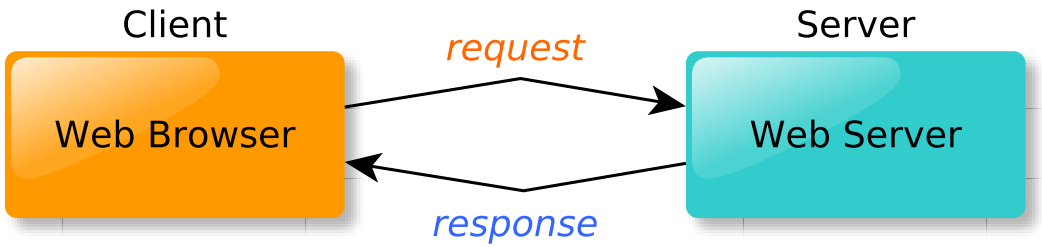
- Uniform Resource Identifier (URI)
  - URL, URN
- `<scheme name>:<hierarchical path> [?<query>] [#<fragment>]`

Basics (Technical), cont.



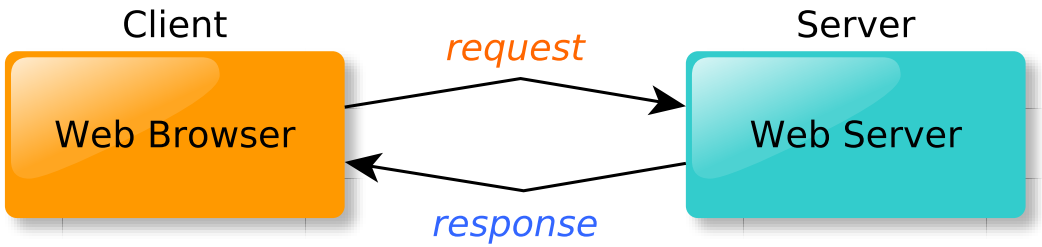
- Uniform Resource Identifier (URI)
  - URL, URN
- `<scheme name>:<hierarchical path> [?<query>] [#<fragment>]`
- Web Browser
  - GET
  - `http://example.com/path?param1=value1&p2=v2`

Basics (Technical), cont.



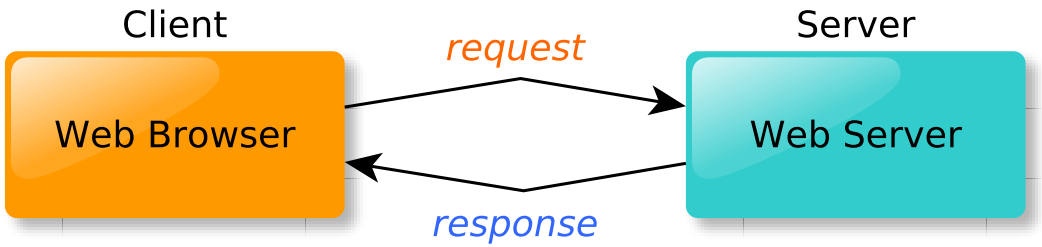
- `response` body could contain anything

Basics (Technical), cont.



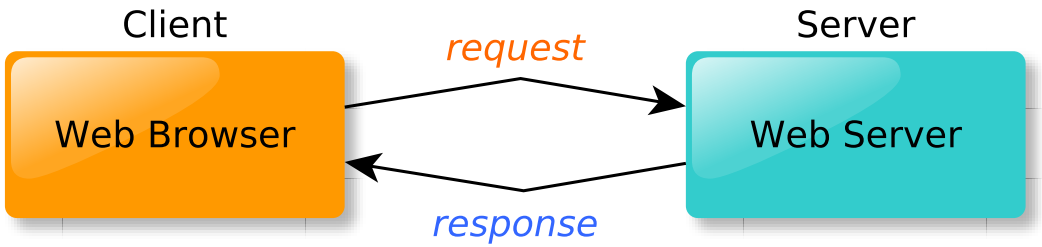
- `response` body could contain anything
- Web Browser is responsible for the presentation
  - typically the browser renders HTML (data)
  - CSS (layout)
  - Javascript (client-side processing)

Basics (Technical), cont.



- `response` body could contain anything
- Web Browser is responsible for the presentation
  - typically the browser renders HTML (data)
  - CSS (layout)
  - Javascript (client-side processing)
- We will primarily deal with the server side

Basics (Technical), cont.



- `response` body could contain anything
- Web Browser is responsible for the presentation
  - typically the browser renders HTML (data)
  - CSS (layout)
  - Javascript (client-side processing)
- We will primarily deal with the server side
- More details later during architectural discussions

Software Engineering: Abstraction & Reuse

- Aims to simplify (for humans)

Software Engineering: Abstraction & Reuse

- Aims to simplify (for humans)
- Early assembly programming
  - sequence of machine calls

Software Engineering: Abstraction & Reuse

- Aims to simplify (for humans)
- Early assembly programming
  - sequence of machine calls
- Procedural programming
  - Structures (global)
  - Procedures/functions
  - Libraries
  - C, Pascal, Basic, Modula, ...

Software Engineering: Abstraction & Reuse

- Aims to simplify (for humans)
- Early assembly programming
  - sequence of machine calls
- Procedural programming
  - Structures (global)
  - Procedures/functions
  - Libraries
  - C, Pascal, Basic, Modula, ...
- Object-oriented programming
  - Encapsulation (inheritance/polymorphism)
  - Interfaces
  - Powerful class libraries
  - Java, C++, C#, Python, *Ruby*, ...

Software Engineering: Abstraction & Reuse, cont.

- Component-*oriented programming*
  - Defined rules and contracts for deployment and reuse
  - Potentially distributed
  - CORBA, COM/DCOM, EJB, CCA, ...

Software Engineering: Abstraction & Reuse, cont.

- Component-*oriented programming*
  - Defined rules and contracts for deployment and reuse
  - Potentially distributed
  - CORBA, COM/DCOM, EJB, CCA, ...
- Service-*oriented programming*
  - Service/contract-based abstractions
  - Platform/OS independent, language independent
  - Naturally distributed
  - Autonomous (independence in 3rd party sense)

Software Engineering: Abstraction & Reuse, cont.

- Component-*oriented programming*
  - Defined rules and contracts for deployment and reuse
  - Potentially distributed
  - CORBA, COM/DCOM, EJB, CCA, ...
- Service-*oriented programming*
  - Service/contract-based abstractions
  - Platform/OS independent, language independent
  - Naturally distributed
  - Autonomous (independence in 3rd party sense)
- Web Services
  - Design focus is on services interface
  - Loosely coupled (distributed) applications
    - Integration at the interface (contract) level
    - ...no implementation dependencies

Web Services

- WWW originally designed for people to share information
  - static content

Web Services

- WWW originally designed for people to share information
  - static content
- Since early days: people have been using HTML forms as interfaces to access programs (CGI)
  - dynamic content

Web Services

- WWW originally designed for people to share information
  - static content
- Since early days: people have been using HTML forms as interfaces to access programs (CGI)
  - dynamic content
- More recently: machine to machine interaction

Web Services

- WWW originally designed for people to share information
  - static content
- Since early days: people have been using HTML forms as interfaces to access programs (CGI)
  - dynamic content
- More recently: machine to machine interaction

"Web Service; just a Web page that is meant to be consumed by an autonomous program as opposed to a human."

— me :)

Web Services, cont.

- are application components
  - basic
  - part of very complex business and scientific processes

Web Services, cont.

- are application components
  - basic
  - part of very complex business and scientific processes
- are distributed

Web Services, cont.

- are application components
  - basic
  - part of very complex business and scientific processes
- are distributed
- communicate using open protocols
  - HTTP, SOAP, JSON, XML, ...

Web Services, cont.

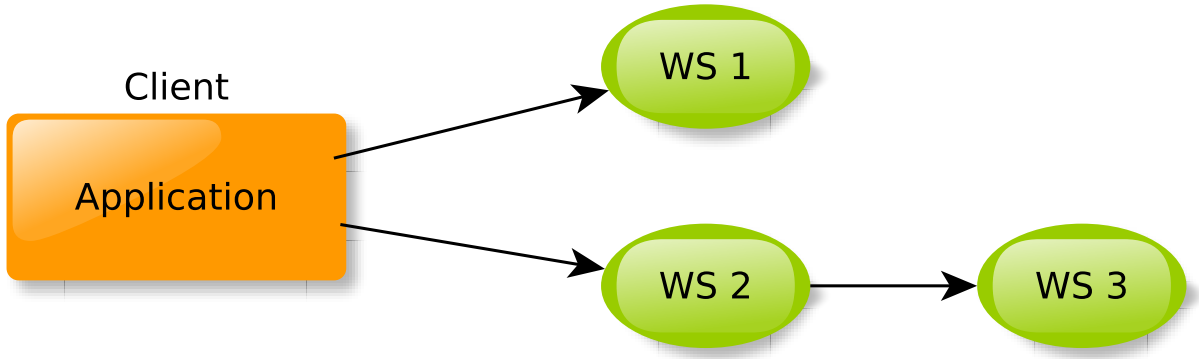
- are application components
  - basic
  - part of very complex business and scientific processes
- are distributed
- communicate using open protocols
  - HTTP, SOAP, JSON, XML, ...
- are platform neutral

Web Services, cont.

- are application components
  - basic
  - part of very complex business and scientific processes
- are distributed
- communicate using open protocols
  - HTTP, SOAP, JSON, XML, ...
- are platform neutral
- can be implemented in different ways
  - HTTP, REST or SOAP ?

Web Services, cont.

- are application components
  - basic
  - part of very complex business and scientific processes
- are distributed
- communicate using open protocols
  - HTTP, SOAP, JSON, XML, ...
- are platform neutral
- can be implemented in different ways
  - HTTP, REST or SOAP ?



Web Applications

- Web site
  - only static content?
  - primarily informational (cnn.com) ?

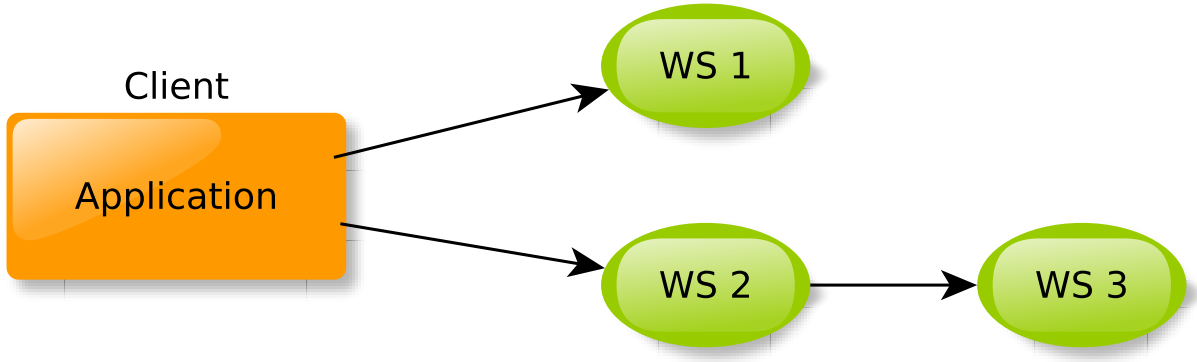
Web Applications

- Web site
  - only static content?
  - primarily informational (cnn.com) ?
- Web application
  - allows user actions (gmail.com) ?
  - allows login (personalisation, state, ...) ?

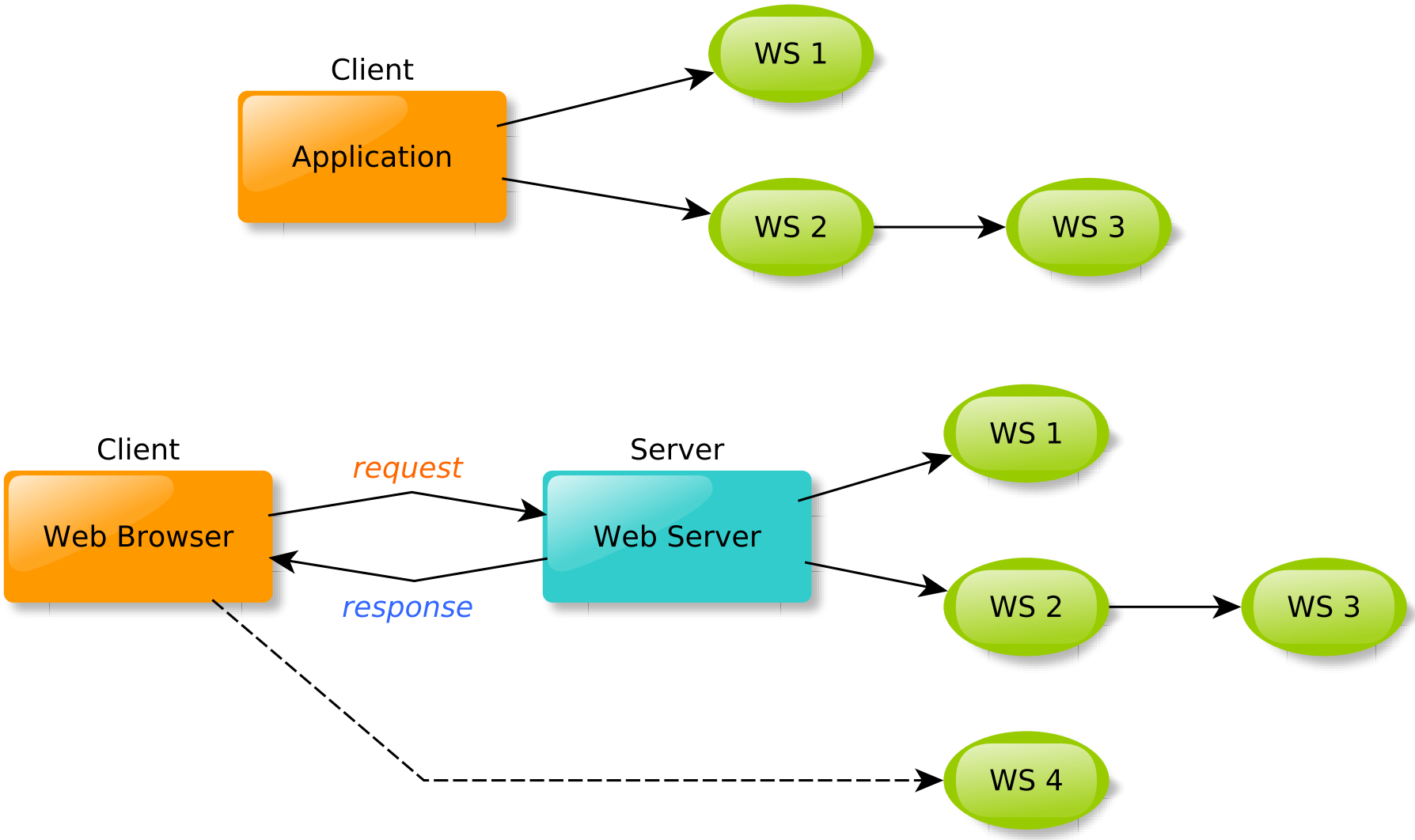
Web Applications

- Web site
  - only static content?
  - primarily informational (cnn.com) ?
- Web application
  - allows user actions (gmail.com) ?
  - allows login (personalisation, state, ...) ?
- Difference is lost nowadays

Web Applications, cont.



Web Applications, cont.



Modern Web Application Characteristics

- Constant evolution

Modern Web Application Characteristics

- Constant evolution
- Huge target group

Modern Web Application Characteristics

- Constant evolution
- Huge target group
- High-quality UI

Modern Web Application Characteristics

- Constant evolution
- Huge target group
- High-quality UI
- Compressed development schedule

Modern Web Application Characteristics

- Constant evolution
- Huge target group
- High-quality UI
- Compressed development schedule
- Major impact of possible failure

Modern Web Application Characteristics

- Constant evolution
- Huge target group
- High-quality UI
- Compressed development schedule
- Major impact of possible failure
- Rapid technological changes

Modern Web Application Characteristics

- Constant evolution
- Huge target group
- High-quality UI
- Compressed development schedule
- Major impact of possible failure
- Rapid technological changes
- Integration of heterogeneous technologies

Modern Web Application Characteristics

- Constant evolution
- Huge target group
- High-quality UI
- Compressed development schedule
- Major impact of possible failure
- Rapid technological changes
- Integration of heterogeneous technologies
- Need to support many platforms/browsers

Modern Web Application Characteristics

- Constant evolution
- Huge target group
- High-quality UI
- Compressed development schedule
- Major impact of possible failure
- Rapid technological changes
- Integration of heterogeneous technologies
- Need to support many platforms/browsers
- Security and privacy

Web Application Frameworks (Ruby)

- Software frameworks
  - Dynamic websites
  - Web Applications
  - Web Services
  - Provide: security, database access, logging, sessions, ...

Web Application Frameworks (Ruby)

- Software frameworks
  - Dynamic websites
  - Web Applications
  - Web Services
  - Provide: security, database access, logging, sessions, ...
- General Purpose frameworks
  - Sinatra
  - Ruby on Rails

Web Application Frameworks (Ruby)

- Software frameworks
  - Dynamic websites
  - Web Applications
  - Web Services
  - Provide: security, database access, logging, sessions, ...
- General Purpose frameworks
  - Sinatra
  - Ruby on Rails
- Wikis
  - Instiki

Web Application Frameworks (Ruby)

- Software frameworks
  - Dynamic websites
  - Web Applications
  - Web Services
  - Provide: security, database access, logging, sessions, ...
- General Purpose frameworks
  - Sinatra
  - Ruby on Rails
- Wikis
  - Instiki
- e-Commerce
  - Spree
- ...

Coursework

- Coursework 1