Introduction to Java EE

The Web Tier

JSF

Enterprise Applications

- “Large-scale, multi-tiered, scalable, reliable, and secure network applications” (Oracle, 2012)

- “The benefits of an enterprise application are helpful, even essential, for individual developers and small organizations in an increasingly networked world.” (Oracle, 2012)

- “The features that make enterprise applications powerful, like security and reliability, often make these applications complex.” (Oracle, 2012)

Java Platform, Enterprise Edition

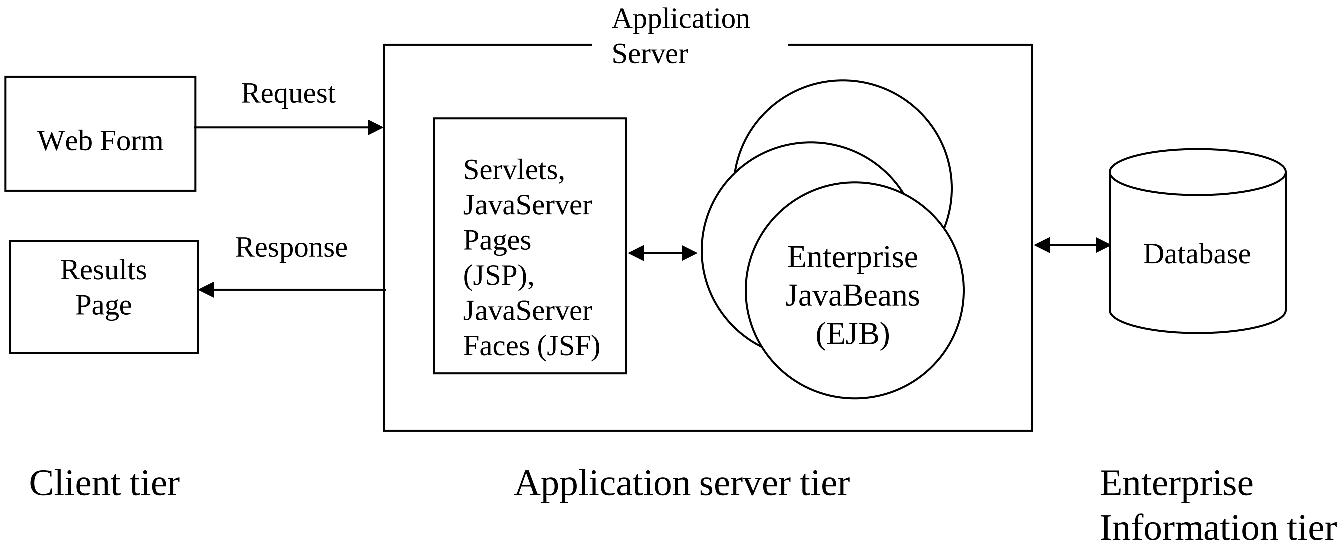
- API and run-time environment for developing enterprise applications
- Designed to solve the problems of large (and small) organisations
  - “Productivity gains through the modern Java EE programming model centered on annotations, POJOs and zero-configuration” (Wilson, 2104)
- “The Java EE platform is designed to reduce the complexity of enterprise application development … [allowing] developers to concentrate on functionality” (Oracle, 2012)

Java Platform, Enterprise Edition

- Multi-tiered, distributed application model
- Component-based
- Unified security model
- Flexible transaction control
- Web services support
- Open systems development - not tied into one vendor

Multi-tiered Distributed Application Model

- Generally regarded as three-tier systems
  - Client tier, Application server tier, Enterprise Information tier


Application Server
Web Form
Request
Results Page
Response
Servlets, JavaServer Pages (JSP), JavaServer Faces (JSF)
Enterprise JavaBeans (EJB)
Database
Client tier
Application server tier
Enterprise Information tier

Java EE is component-based

- A component is a self-contained functional software unit that is assembled into a Java EE application
  - with its related classes and files
  - communicates with other components
- Some components run on the client
  - e.g. applications and applets
- Some components run on the server
  - Servlet, JSP, and JSF are Web components
    - JSP is now deprecated (in favour of JSF) but still widely used (e.g. Blackboard)
  - Enterprise JavaBeans (EJB) are Business components
    - EJBs are also known as enterprise beans

Client Tier Components

- Web client
  - Dynamic web pages generated by components in the Web Tier
  - The web browser that renders these pages

- Applets
  - Small Java applications embedded in a web page
  - Executed in the JVM installed in the web browser (Java plug-in)
  - Java plug-in is to be deprecated (BBC, 2016)
  - Use Java Web Start instead
    - “a framework … that allows users to start [Java applications] directly from the Internet using a web browser.” (Wikipedia, 2017)

Client Tier Components

- Application client
  - Runs on a client machine
  - Provides a way for users to handle tasks that require a richer user interface than can be provided by a mark-up language
  - Typically has a GUI created from Java Swing or AWT APIs
  - Can be distributed via the Web using Java Web Start
  - A command-line interface is also possible

Application Server Tier Components

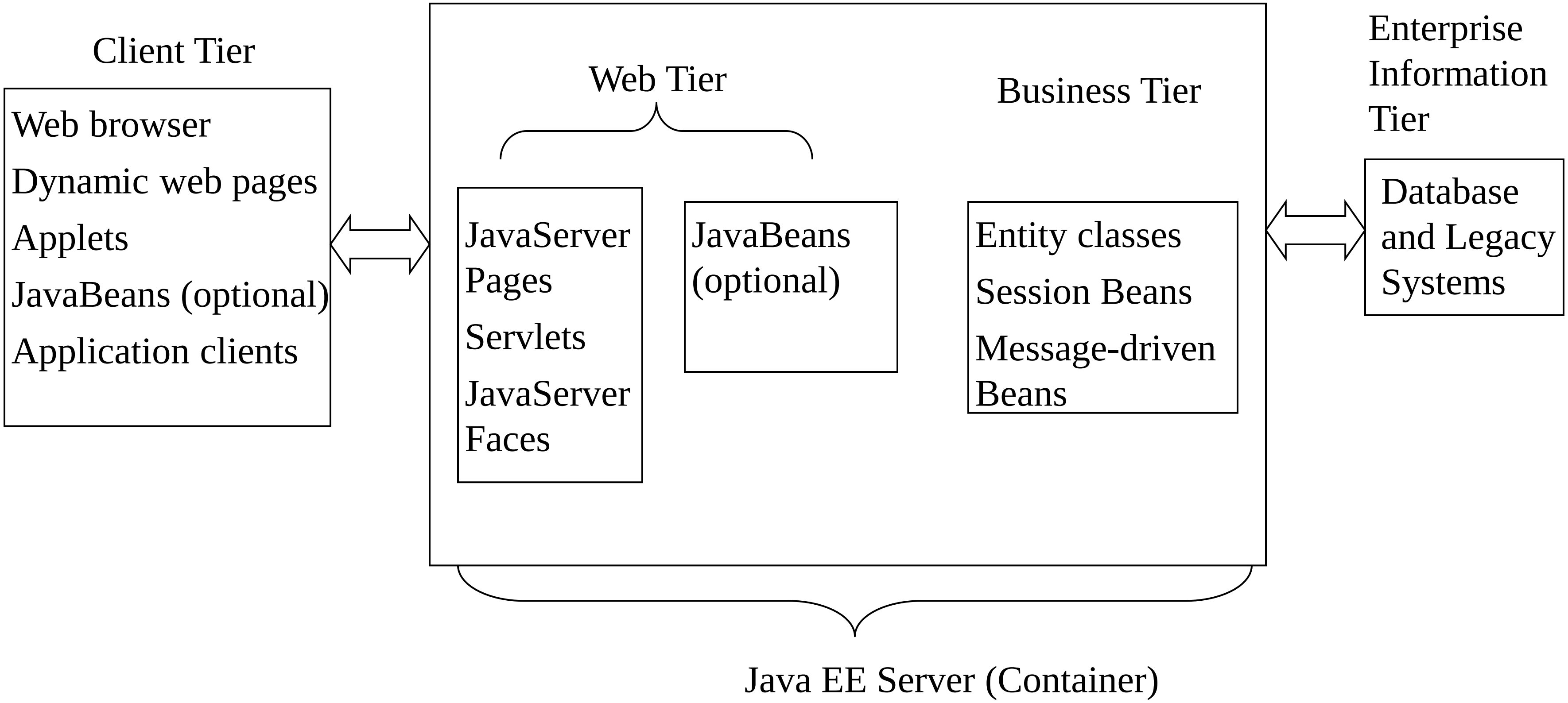
- Java EE Web Components
  - Servlets, JSP pages or JavaServer Faces
    - Servlets are Java programming language classes that dynamically process requests and construct responses
    - JavaServer Pages are text-based documents that execute as servlets but allow a more natural approach to creating static content
    - JavaServer Faces is a server-side framework for building web applications

Application Server Tier Components

- JavaBeans Components
  - Applications can include JavaBeans
    - POJOs with properties, get- and set- methods
    - Not considered Java EE components by the Java EE specification

- Business Components (Enterprise JavaBeans)
  - Handle the logic that solves or meets the needs of a particular business domain, such as banking

The Different Components



Client Tier

Web browser

Dynamic web pages

Applets

JavaBeans (optional)

Application clients

Web Tier

JavaServer Pages

Servlets

JavaServer Faces

JavaBeans (optional)

Business Tier

Entity classes

Session Beans

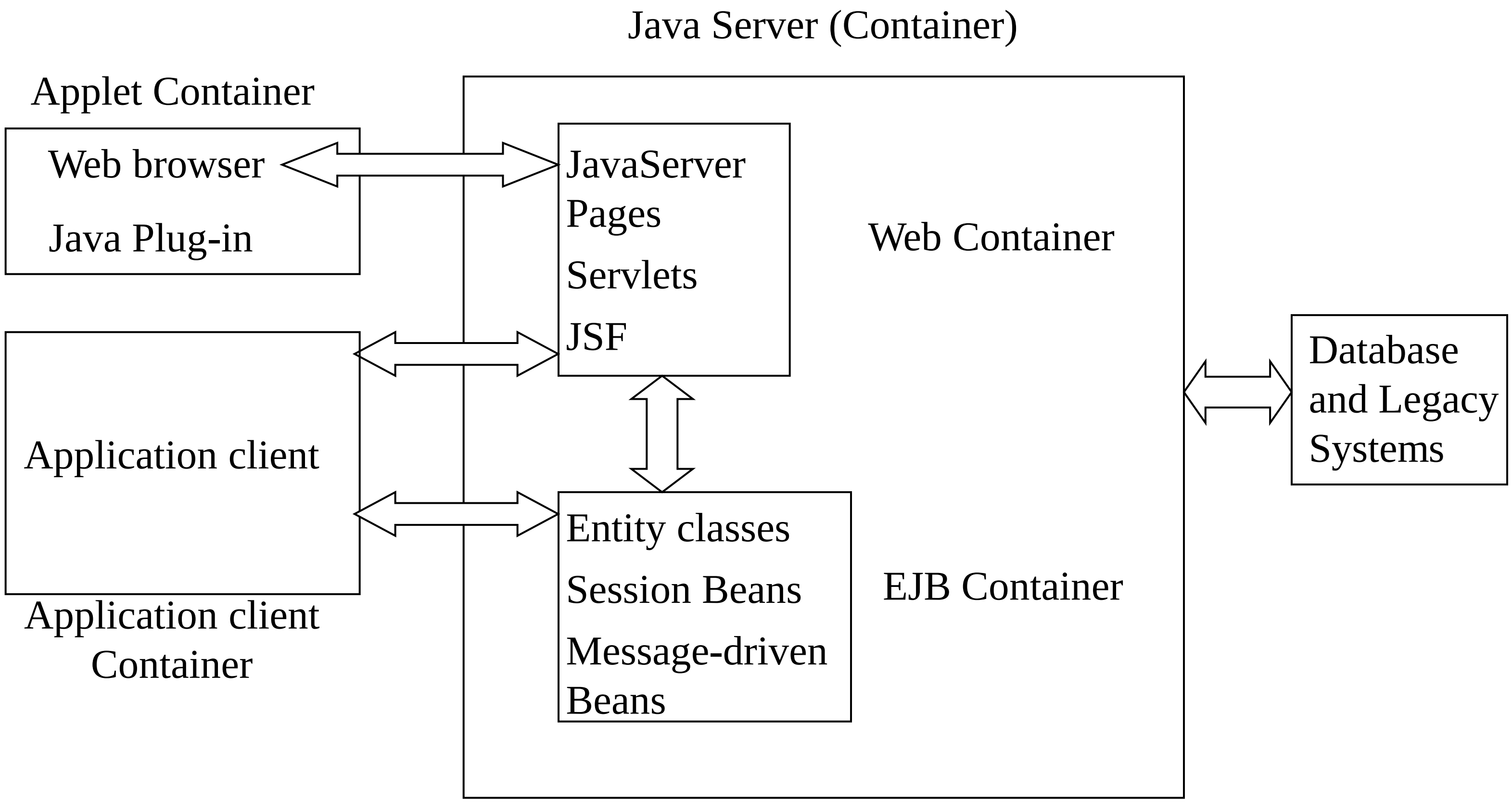
Message-driven Beans

Enterprise Information Tier

Database and Legacy Systems

Java EE Server (Container)

The Different Containers



Java Server (Container)

Applet Container

Web browser

Java Plug-in

JavaServer Pages

Servlets

JSF

Web Container

Application client

Application client Container

Entity classes

Session Beans

Message-driven Beans

EJB Container

Database and Legacy Systems

Containers

- A container is a runtime service
  - provides the interface between components and the low-level functionality that supports the components

- A container manages components that have been developed according to the API specifications

- We shall use MySQL or Oracle’s GlassFish Server as our Java EE container
  - It provides both a web container (for servlets, JSP, and JSF)
  - and an EJB container (for enterprise beans)

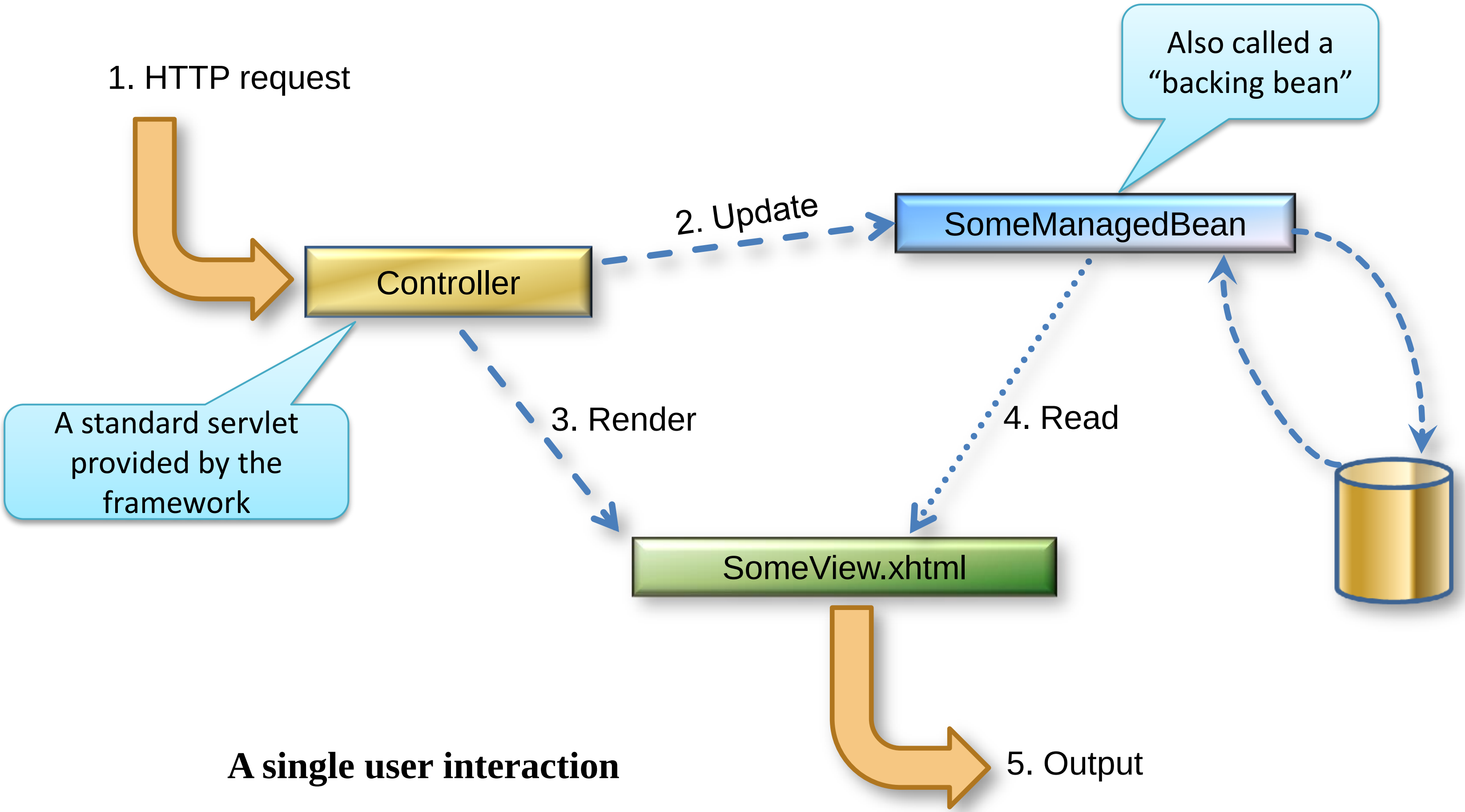
Server-side Technologies

- Servlets, JSP and JSF are server-side technologies
  - accept HTTP requests from an HTML form
  - return responses in the form of HTML, XHTML or XML

- At runtime, servlets and JSP are equivalent
  - JSP is automatically converted to a servlet

- JSF is a framework with a distinctive request-response life-cycle
  - We write components that fit into that life-cycle

JSF components



A single user interaction

JSF request-response life-cycle

- Implemented by the standard JSF servlet

- For a given HTTP request for a view…
  - Generate a tree of components in the submitted form
    - Populate with request parameters & previous state
    - Perform conversions and validation

  - Update managed bean components
    - Copy values from component tree to bean properties

  - Render response
    - Output the view using managed bean properties
    - Save state

JSF example

- Make your own notes about the code
  - Download wk2-01_lecProj.zip from the resource link

Statelessness and Sessions

- HTTP is a stateless protocol
  - Each request from a client is treated in isolation from all other requests from the same client
  - there is no relationship between the different requests
- A protocol is stateful if the response to a given request may depend
  - not only on the current request
  - but also on the outcome of previous requests

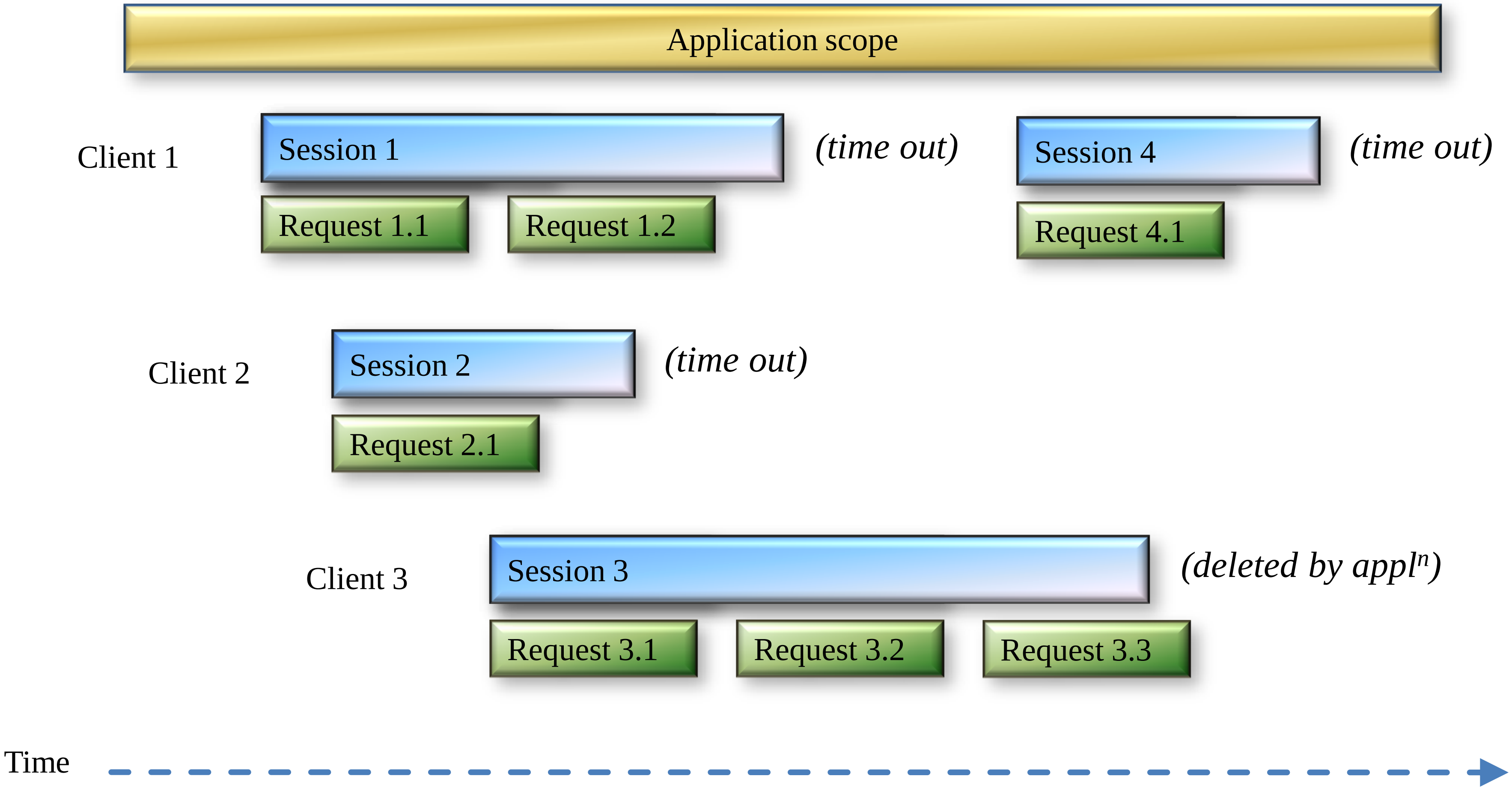
Statelessness and Sessions

- Stateful is important
  - With on-line banking you only need to log-in once
    - Checking balances, viewing statements, or transferring money doesn’t require log-in for every request
  - Internet shopping with a shopping cart

- Session
  - The server can identify a series of requests from a single client as a single working session

- State
  - The server can remember information related to previous requests in the same session

Illustration of scope duration



References

- BBC (2016) *Java browser plug-in to be retired, says Oracle* [Online] Available from: http://www.bbc.co.uk/news/technology-35427685 [Accessed: 13/01/2017]
- Oracle (2012) *Your First Cup: An Introduction to the Java EE Platform* [Online] Available from: http://docs.oracle.com/javaee/6/firstcup/doc/firstcup.pdf [Accessed: 13/01/2017]
- Wilson, K. (2014) *Java EE/Glassfish adoption story* [Online] Available from: https://blogs.oracle.com/stories/entry/java_ee_glassfish_adoption_story [Accessed: 13/01/2017]
- Wikipedia (2017) *Java Web Start* [Online] Available from: https://en.wikipedia.org/wiki/Java_Web_Start [Accessed: 13/01/2017]