Edinburgh Napier
UNIVERSITY

ADVANCED WEB TECHNOLOGIES
SET09103
TOPIC #02

Dr Simon Wells
s.wells@napier.ac.uk
http://www.simonwells.org

Edinburgh Napier
UNIVERSITY

OVERVIEW

- Part #1: Web pages & Web sites
- Part #2: Creating routes

Edinburgh Napier UNIVERSITY

TL/DR

The Web is an existence proof for a massively scalable distributed information system

Edinburgh Napier
UNIVERSITY

PART # I

Edinburgh Napier
UNIVERSITY

WHAT IS A WEB PAGE?

Edinburgh Napier
UNIVERSITY

A WEB PAGE IS…

- A file stored on a Web server that is made available (served) to a client in response to a request.

- The file contains HTML code

Edinburgh Napier
UNIVERSITY

WHAT IS A WEB SITE?

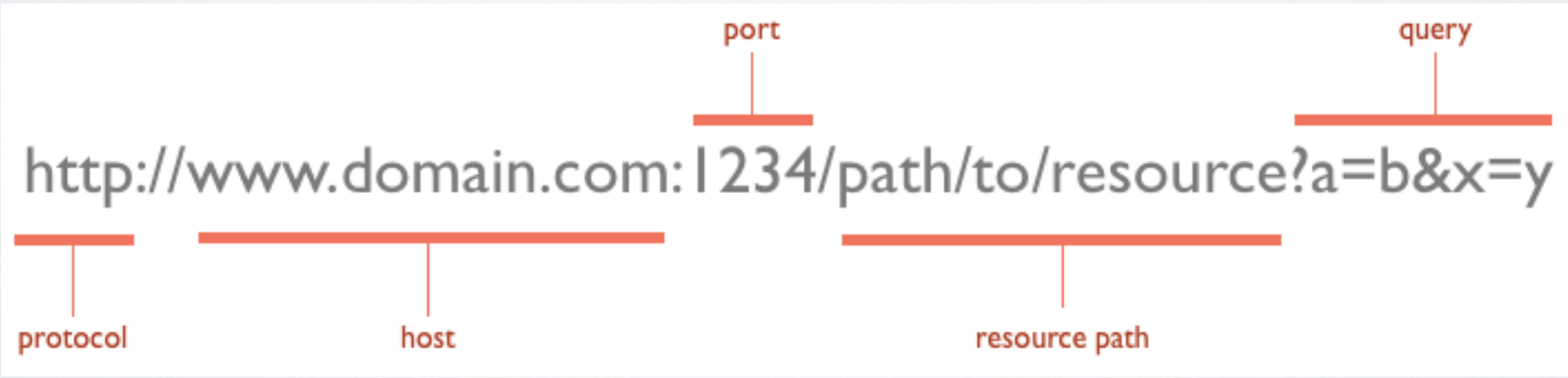
Edinburgh Napier
UNIVERSITY

A WEB SITE IS…

- Trivially: A collection containing at least one web page.
- More accurately: Websites are collections of resources.
- Where a resource is a web page (e.g. An HTML pages).
- One web site might have many web pages.

REVISITING URLS

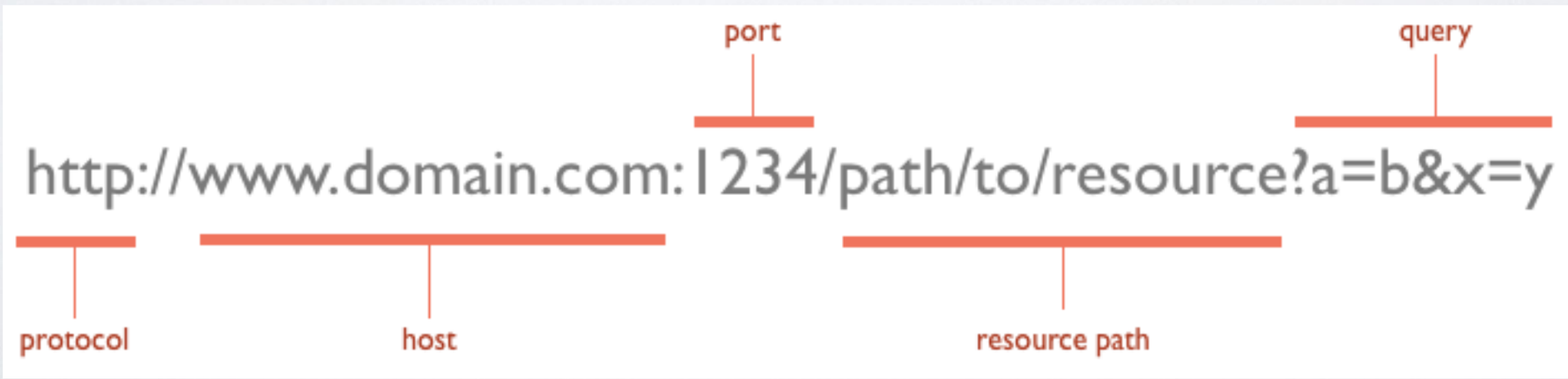
- The request message is at the heart of communications via the web
- A request will be sent to an HTTP server with respect to a URL
- The URL has structure (which we will exploit later)
- **Protocol** is usually http (but can also be https)
- **Port** defaults to 80 but can be set explicitly in the URL
- **Resource Path** is the local path to the resource on the server



Edinburgh Napier UNIVERSITY

URLS IN FLASK

- Flask calls the path to a resource/page a **route**.
- A simple page in Flask is a Python function that returns a String containing the content for the browser to display.
- A decorator is used to pair the function with a route description.
- *Later we will also add query parameters to the end of the address.*
- *We'll also use actual html files and templates, but let's walk before we run.*



Edinburgh Napier
UNIVERSITY

HELLO WORLD REVISITED

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "<p>Hello World!</p>"
```

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def home():
    return “<p>An app that greets people</p>”
```

Edinburgh Napier UNIVERSITY

```
@app.route(“/hello/")
def hello():
    return "<p>Hello World!</p>"

@app.route(“/goodbye/")
def goodbye():
    return “<p>Goodbye Cruel World!</p>"
```

Edinburgh Napier
UNIVERSITY

WHAT IS THE WEB?

THE WEB IS…

- An inter-connected collection of Websites.
- Interconnections are formed using hyperlinks (HTML feature):

link text

- Hyperlinks can be:
  - Between pages.
  - Within a page (by using anchors as a place for the hyperlink to point to):
    - Give element an ID then add # followed by the ID to the hyperlink.
- Any page can link to any other (accessible) page.

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def home():
    return ‘’’’’
      <p>An app that greets people:</p>
      <p>Say <a href=‘’/hello/’’>hello</a> or
      <a href=‘’/goodbye/’’>goodbye</a></p>
    ’’’’’’

@app.route(‘’/hello/")
def hello():
    return "<p>Hello World!</p>"

@app.route(‘’/goodbye/")
def goodbye():
    return ‘’<p>Goodbye Cruel World!</p>"
```

Edinburgh Napier
UNIVERSITY

Edinburgh Napier UNIVERSITY

ADDRESS HIERARCHIES

- Your site will usually start somewhere:
  - home page (index.html, home.html) - the default that is served when you navigate to the hostname/base address.
- After that you can have as many pages as are needed to fulfill the design goals of your site.
- Some pages within your site will link to other pages (both within and outwith your site).
- Determining which pages to implement and how those pages are organised/ linked together is a design decision that greatly affects navigation and user experience.

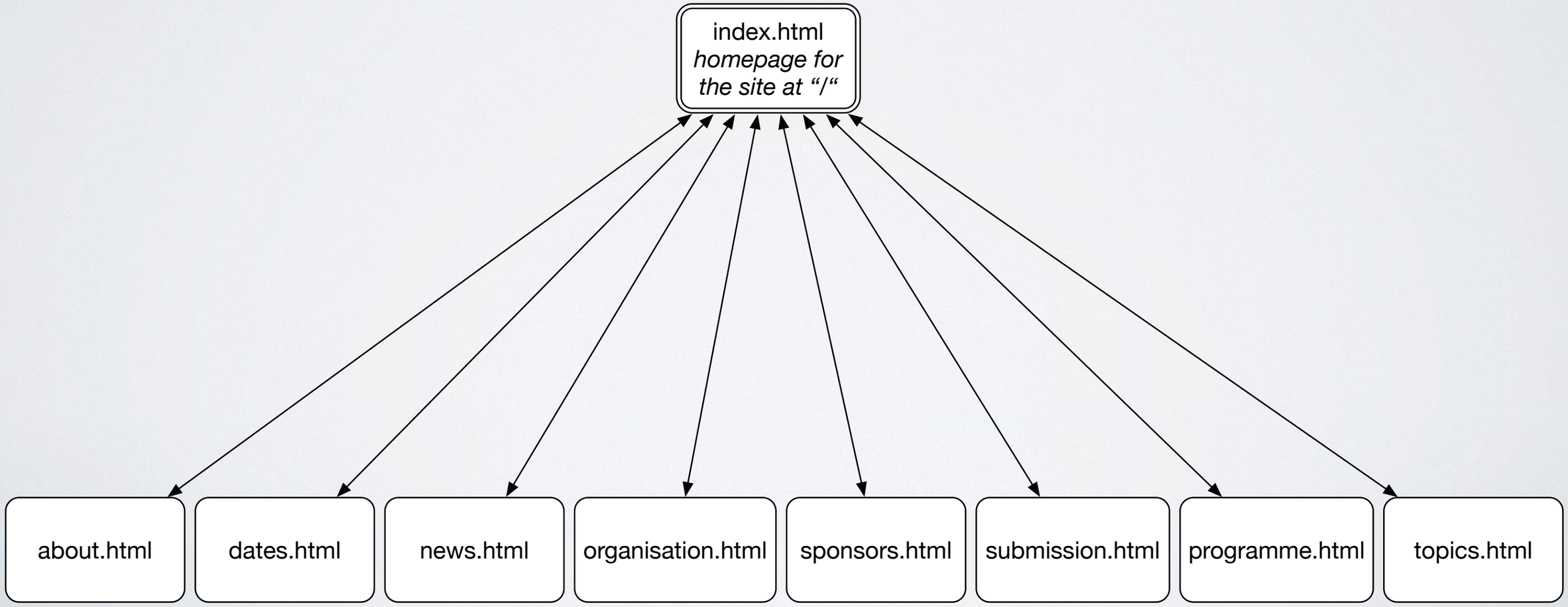
Edinburgh Napier UNIVERSITY

A SIMPLE URL HIERARCHY

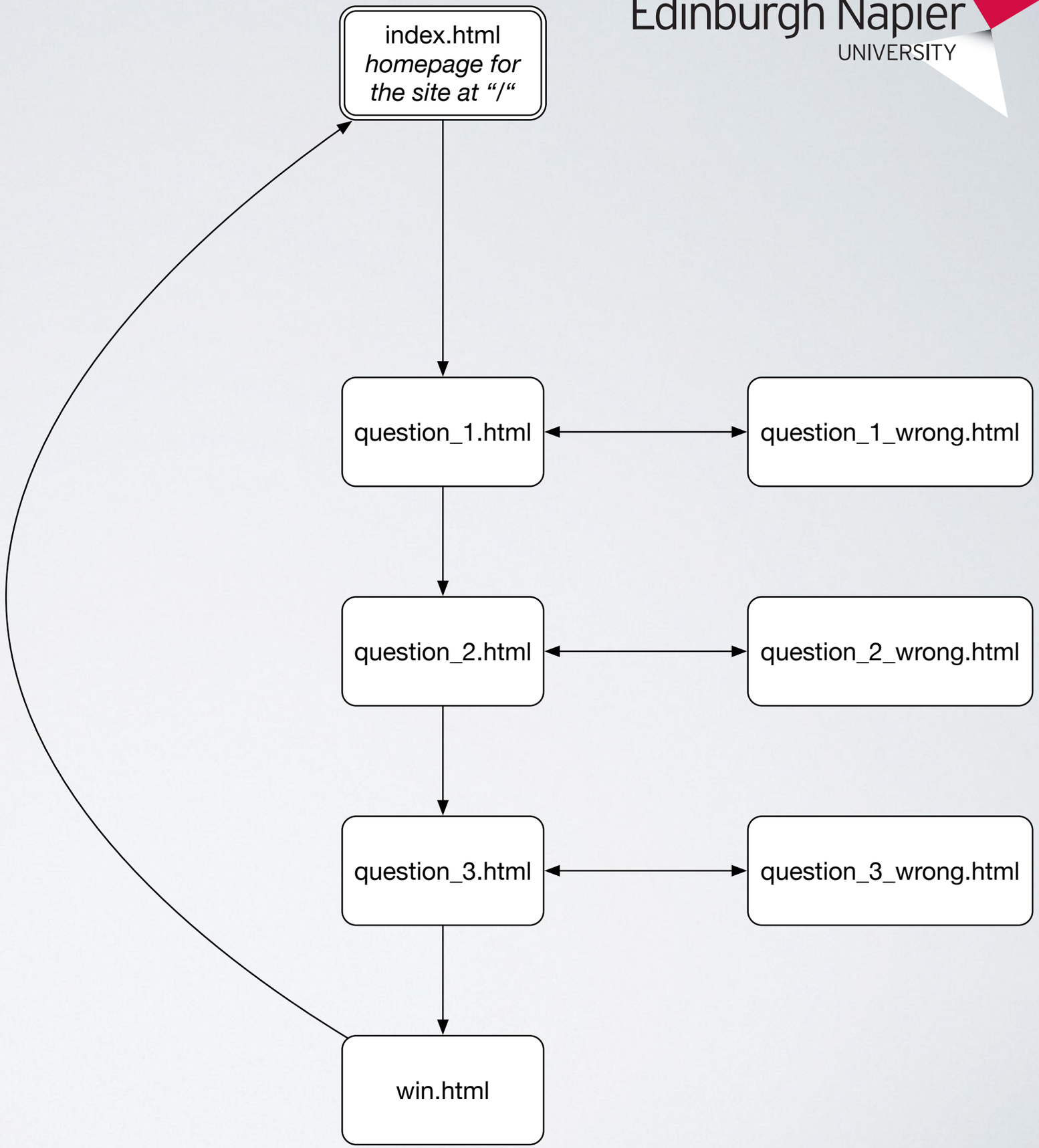
- Address hierarchy for an information website for a research meeting:

https://cmna-workshop.github.io/cmna25/

- Note that many websites are not this simple



- A simple "Quiz" Site.
- Not quite as linear & interconnected.
- Every page still connects to other pages.
- Definitely other ways to accomplish this (which we'll see as we progress).



Edinburgh Napier UNIVERSITY

- We'll return to a similar style of diagram later in the module when we consider templates.

- Many of your pages within a site will have share similar visual design.

- It is sensible to

Edinburgh Napier UNIVERSITY

A QUICK DIVERSION: REST

- **The Web is an existence proof for a massively scalable distributed information system**

- Taking a RESTful approach **affects** (ideally positively) a range of identifiable architectural properties of distributed hypermedia systems:

  - (Perceived) Performance - component interactions can be a dominant factor in user perception of system performance and network efficiency

  - Scalability - Support large numbers of components & interactions between them

  - Simple interfaces

  - (Run-time) modifiability of interfaces

  - Visibility of communication between components

  - Portability of components (move code with data)

  - Reliability - system should be resistant to failure even if components, connectors, or data fail in some way

- We'll return to REST when we consider Web services & API design later…

ARCHITECTURAL CONSTRAINTS

Edinburgh Napier UNIVERSITY

- The architectural constraints are applied with the aim of realising the architectural properties (performance, scalabiliity, reliability, &c.):
  - Client Server, Stateless, Cacheable, Layered System, (optional) code on demand, Uniform Interface
- If a server violates constraints then it is not considered to be RESTful
- But complying with the constraints (even if not working with HTTP) can yield systems that:
  - conform to the REST architectural style, and
  - enables the resulting system to have those desirable properties:

Edinburgh Napier UNIVERSITY

SUMMARY

- At this point we have everything we need to:
  - Dynamically create HTML files (web pages)
  - Collect multiple files together (individual site)
  - Link between files (web site)
  - Essentially: **HTML + Links = Hypermedia**
- NB. Technically not a part of the web as not published anywhere & so nothing can link back to us but this is a mere detail we can address later.
  - Not all uses of Web technologies is for Web sites.
  - A default, cross-platform (G)UI for accessing and sharing information.

Edinburgh Napier
UNIVERSITY

PART #2: PRACTICAL WORK

Edinburgh Napier UNIVERSITY

EXAMPLE #1

- Greetings App:
  - *Demonstrating route implementation.*

EXAMPLE #2

- Super simple Quiz app:
  - Demonstrating page interconnection using hyperlinks.

Edinburgh Napier UNIVERSITY

WRAPPING UP

- Routes.
- HyperLinks.
- Address/URL Hierarchies & their design.
- Underlying theory: REST properties & constraints.