INTRODUCTION TO INTERACTIVE WEB APPLICATIONS 👨‍💻

- Press `Space` to navigate through the slides
- Use `Shit+Space` to go back

MODULE STRUCTURE 📖

- Lecture
    - **Thursday** – 09:00-11:30 Group **A**
    - **Thursday** – 12:00-2:30 Group **B**

- Breakdown of assesments
    - **CA1** in Semester 1 – 25%
    - **CA2** in Semester 2 – 25%
    - **EXAM** – 50%

- 2% penalty per day for late submissions

PLAGIARISM 🚫

- **CCT** is constantly striving to build a culture which values and supports good academic conduct
- Academic dishonesty is **a serious offence** that will not be tolerated
- Students found to be involved in plagiarism will be brought before the **Academic Standards Board**

IN-CLASS TECHNOLOGIES ⚙️

- **GitHub** – http://github.com (http://github.com)
- **Slack** – http://slack.com (http://slack.com)
- **Jupyter Notebook** – http://jupyter.com (http://jupyter.com)
- **Binder** – http://mybinder.org (http://mybinder.org)
- **Moodle** – http://moodle.cct.ie (http://moodle.cct.ie)

A BRIEF HISTORY OF COMPUTING 

- **pre-1960s:** IBM, first electronic computers
- **1960s:** Intel, file system, mainframes
- **1970s:** Microsoft, Apple, Unix, first personal computers
- **1980s:** Era of PCs, MS-DOS, Internet
- **1990-2000s:** Rise of WWW, client-server, Web 2.0

SERVER-SIDE SCRIPTING 🖥️

- Access to server-side environment
- Database interactions
- Access to server file-system
- Authentication / Authorisation
- Process requests and construct appropriate responses

- **PHP, Perl, Ruby, Node.js and other solutions**

CLIENT-SIDE SCRIPTING 

- Access to client-side environment – typically a browser
- Manipulate client-side environment
- Respond to client-side events
- Perform client-side validation of data
- Security restrictions in place

- **JavaScript and its frameworks**

DATA EXCHANGE ♻️

There are four central problems in data management:

- Capture
- Storage
- Retrieval
- Exchange of data

RDF, XML, Atom, JSON, YAML, REBOL, Gellish

EXTENSIBLE MARKUP LANGUAGE 🗒️

The popularity of **XML** for data exchange on the **World Wide Web** has several reasons:

- Closely related to the preexisting standards **Standard Generalised Markup Language (SGML)** and **Hypertext Markup Language (HTML)**
- **XHTML** has been defined as a format that is formal **XML**, but understood correctly by most **HTML** parsers
- The design goals of **XML** emphasise simplicity, generality, and usability across the Internet

- **XML** stands for eXtensible Markup Language
- **XML** is a mark-up language much like HTML
- **XML** was designed to carry data, not to display data
- **XML** tags are not predefined – **you define your own tags!**
- **XML** is designed to be self-descriptive
- **XML** is a W3C Recommendation

- Maybe a little hard to conceptualise, but **XML *does not DO anything***.
- **XML** was created to structure, store, and transport information.

In [29]:
```
from display_xml import XML
XML('<note><to>Tove</to><from>Jani</from><heading>Reminder</heading><body>Don&apo
s;t forget me this weekend!</body></note>')
```

Out[29]:
```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- The note above is quite self descriptive
- It has sender and receiver information, it also has a heading and a message body
- But still, this XML document does **not do anything**
- It is just information wrapped in tags
- Someone must write a piece of software to send, receive or display it.

XML 🕵️

- **eXtensible Mark-up Language (XML)** is a mark-up language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable
- Many **application programming interfaces (APIs)** have been developed for software developers to use to process **XML** data, and several schema systems exist to aid in the definition of **XML-based languages**.

WHY XML IS NEEDED? 🤷‍♀️

- **XML** provides a robust and durable format for information storage and transmission
- Displaying the same data in multiple formats
- Information tailored to fit the needs of the reader
- Moving the processing load from the web server to the client

BENEFITS OF XML 🌟

- Programmers structure data in an infinite variety of ways
- With every new way of structuring data comes a new methodology for extracting the information we need
- If the data changes, the methodologies also have to change, and testing and tweaking begin again
- With **XML** there is a standardised way to get the information, no matter how we structure it

- We have control over how we structure our **XML** files
- The data can be marked-up in any way we choose
- Agreed standard formats to perform common tasks can make data exchange between applications easier
    - Scalable vector graphics **(SVG)** describes 2D graphics
    - **MathML** used to describe mathematics

HTML, XML, CSS 🌐

- **HTML** for display
- **XML** for data exchange
- **HTML** is designed for a specific application and to convey information visually in a browser
- Because **HTML** has a specific application it is also restricted by having a specific set of finite markup constructs that are used to create an **HTML** document
- **CSS** is used with HTML and XML to display the contents of a document in a clear and precise manner.

SUMMARY 🕶

- Evolution of computing
- Server-side & Client-side
- Data exchange and XML
- Uses and benefits of XML
- HTML, XML and CSS