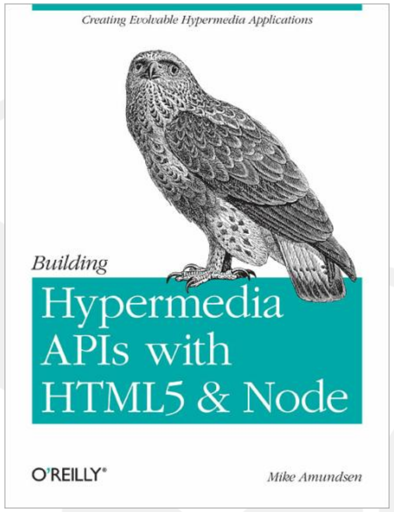
Mike Amundsen

The Costs and Benefits of Building Hypermedia APIs (with Node.js)


Creating Evolvable Hypermedia Applications
Building
Hypermedia APIs with HTML5 & Node
O'REILLY
Mike Amundsen

Mike Amundsen

- Author
- Presenter
- Software Explorer
- **Principal API Architect**

OK, let’s get started…

Theonia

"a looking at, viewing, beholding"

Theory

"a looking at, viewing, beholding"

Praxis

“doing”

Practice

“doing”

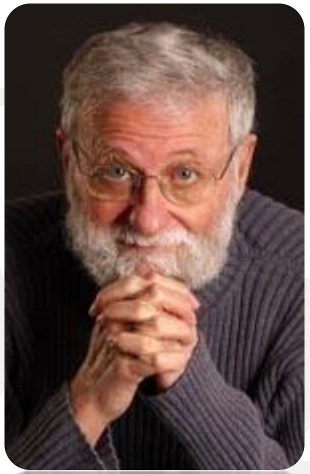
First, a message
from Donald Norman…

Affordances

Affordances

“The value of a well-designed object,
Is when it has such a rich set of affordances,
That the people who use it,
Can do things with it,
That the designer never imagined.”

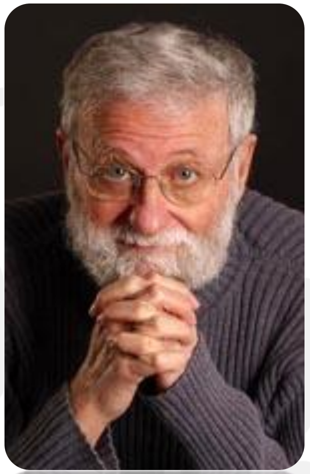
- Donald Norman



Affordances

"The value of a well-designed object,
Is when it has such a rich set of affordances,
That the people who use it,
Can do things with it,
That the designer never imagined."

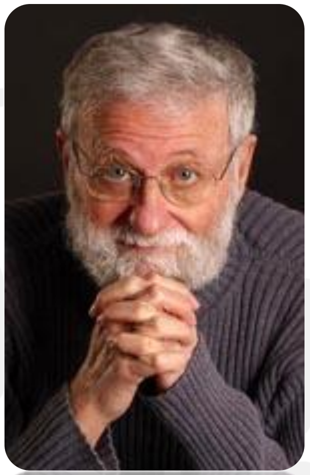
- Donald Norman



Affordances

"The value of a well-designed object,
Is when it has such a rich set of affordances,
That the people who use it,
Can do things with it,
That the designer never imagined."

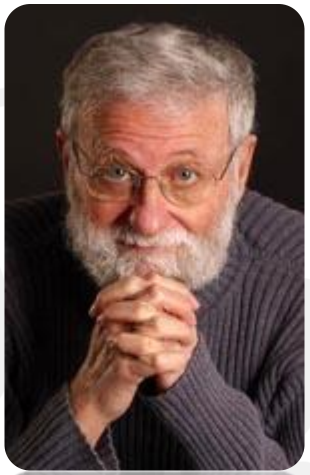
- Donald Norman



Affordances

"The value of a well-designed object,
Is when it has such a rich set of affordances,
That the people who use it,
Can do things with it,
That the designer never imagined."

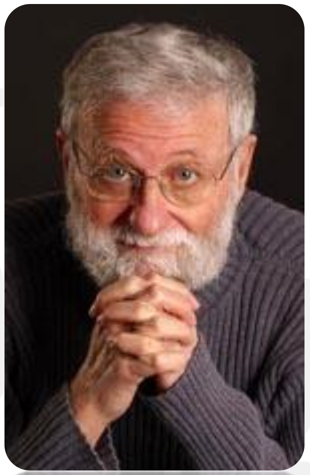
- Donald Norman



Affordances

"The value of a well-designed object,
Is when it has such a rich set of affordances,
That the people who use it,
Can do things with it,
That the designer never imagined."

- Donald Norman



Some background…

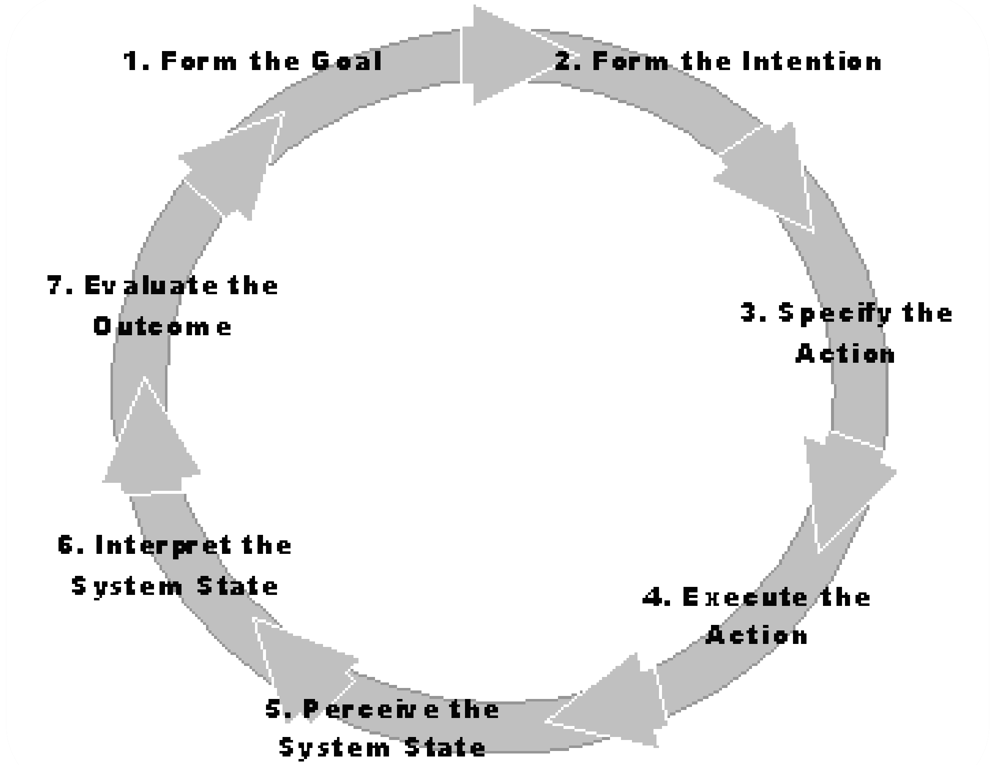
Affordances

- The *foundation* for perception is ambient, ecologically available information.
- **Affordances** are all "action possibilities" latent in the environment.

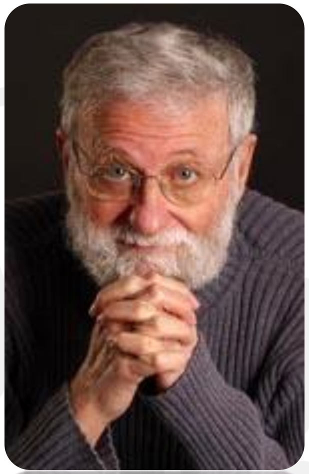
Theory of Affordances, 1979
- James J. Gibson

Affordances

- Seven Stages of Action



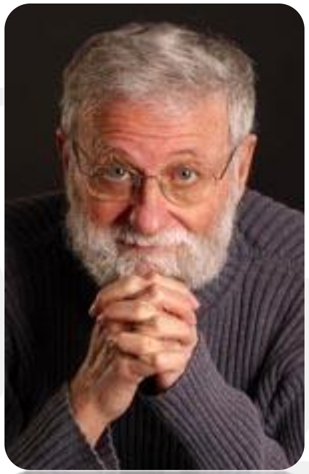
The Design of Everyday Things, 1988
- Donald Norman



Affordances

Knowledge ("head" vs. "world")

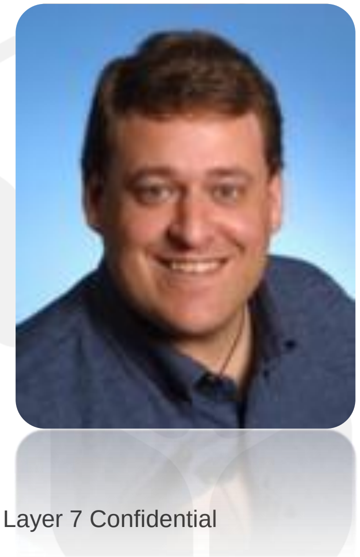
Property	Knowledge in the World	Knowledge in the Head
Learning	Learning not required. Interpretation substitutes for learning. How easy it is to interpret information is the world depends upon how well it exploits natural mappings and constraints.	Requires learning, which can be considerable. Learning is made easier if there is meaning of structure to the material (or if there is a good mental model).
Efficiency of use	Tends to be slowed up by the need to find and interpret the external information.	Can be very efficient
Ease of use at first encounter	High	Low



Affordances

- *"Hypermedia is defined by the presence of application control information embedded within, or as a layer above, the presentation of information" (2001)*
- *"When I say [Hypermedia], I mean the simultaneous presentation of information and controls such that the information becomes the affordance through which the user obtains choices and selects actions" (2008)*

Architectural Styles and the Design of Network-based Software, 2001
- Roy T. Fielding



Affordances

- *"Hypermedia is defined by the presence of application control information embedded within, or as a layer above, the presentation of information“ (2001)*
- *“When I say [Hypermedia], I mean the simultaneous presentation of information and controls such that the information becomes the affordance through which the user obtains choices and selects actions” (2008)*

Architectural Styles and the Design of Network-based Software, 2001
- Roy T. Fielding

Affordances

Affordances make Hypermedia possible.

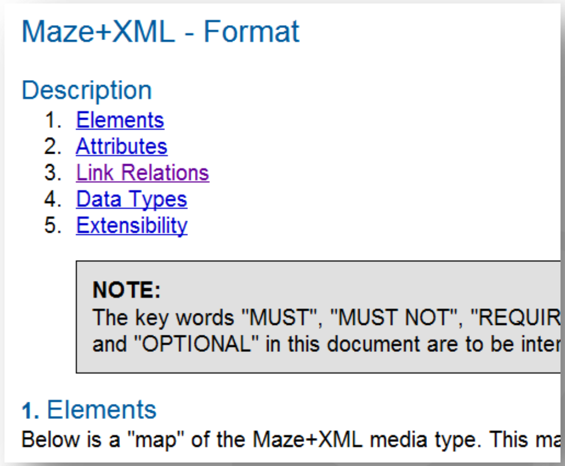
Maze+XML

Design #1:
A big pile of Affordances

Maze+XML

- **Maze+XML** media type
- First design in late 2010, registered w/ IANA 2011
- *"…an XML data format for sharing maze state information between clients and servers. It can be used to implement simple mazes, adventure games, and other related data."*
- *Read-only navigational links*
- *Nine link identifiers:*

  collection, maze, start, exit, current, north, south, east, west



Maze+XML - Format

Description

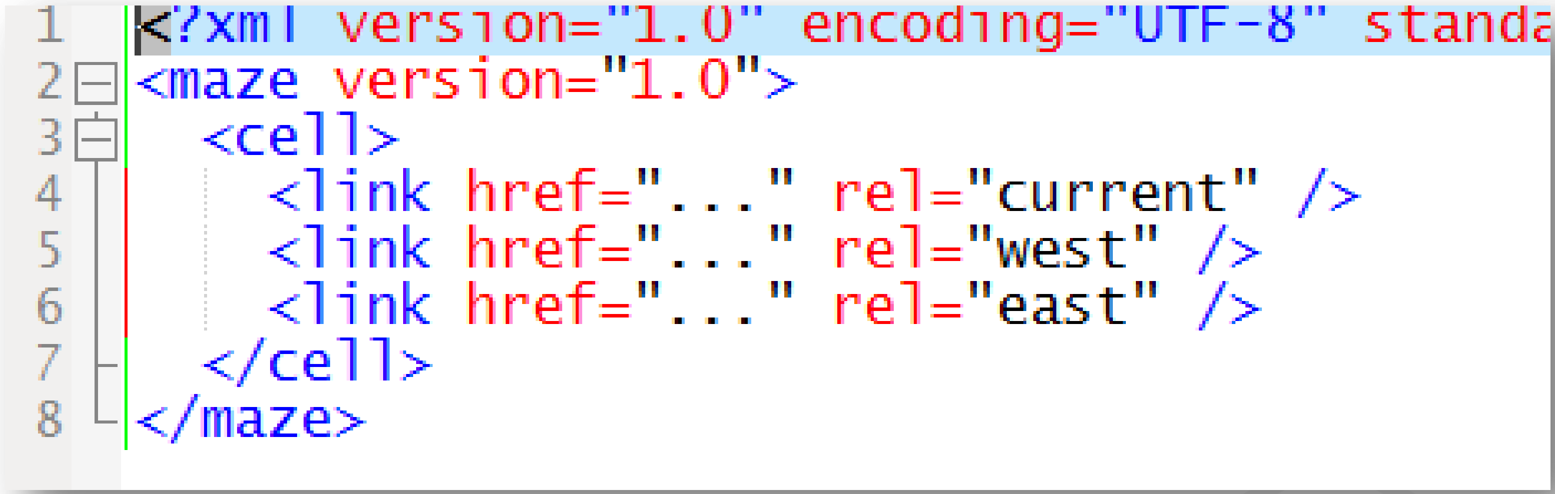
1. Elements
2. Attributes
3. Link Relations
4. Data Types
5. Extensibility

NOTE:
The key words "MUST", "MUST NOT", "REQUIR
and "OPTIONAL" in this document are to be inter

1. Elements
Below is a "map" of the Maze+XML media type. This ma

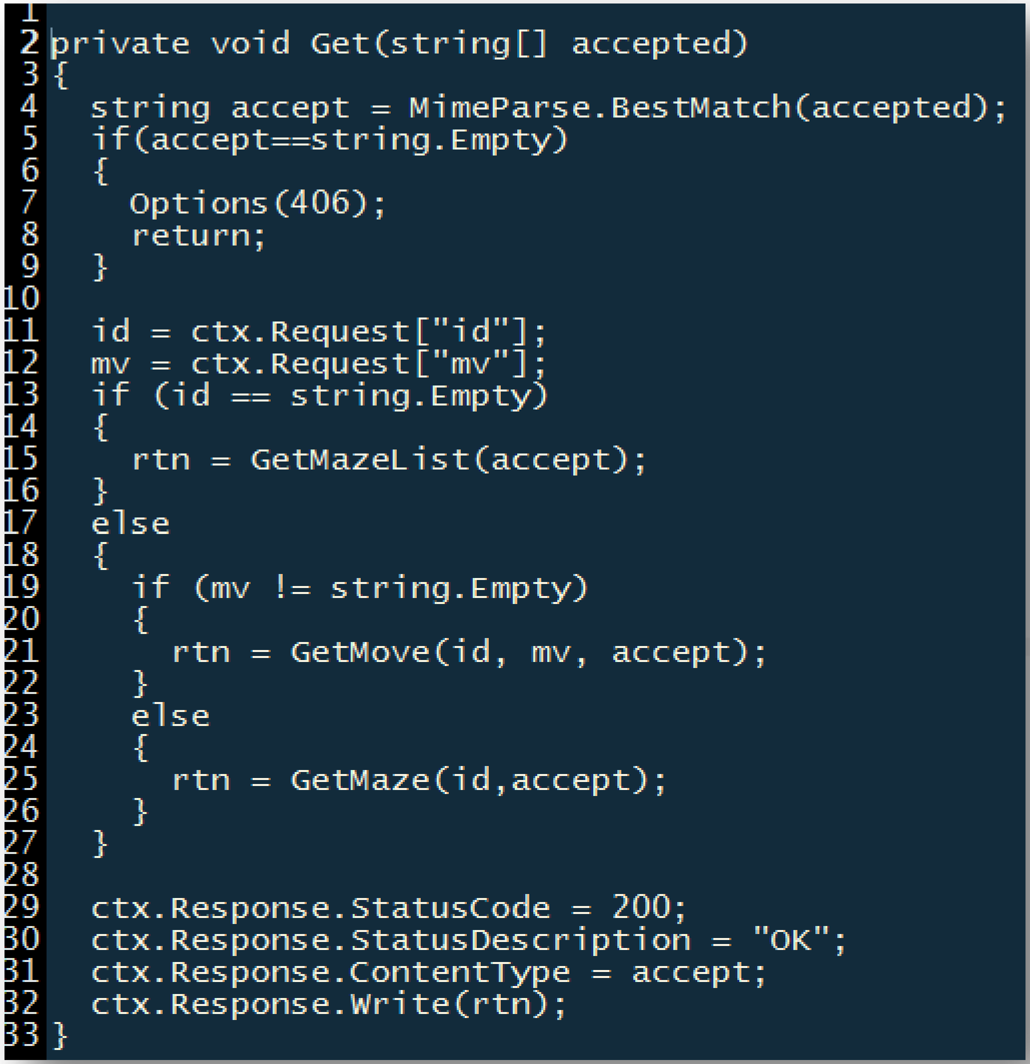
Maze+XML

Message



```
<?xml version="1.0" encoding="UTF-8" standa
<maze version="1.0">
  <cell>
    <link href="..." rel="current" />
    <link href="..." rel="west" />
    <link href="..." rel="east" />
  </cell>
</maze>
```

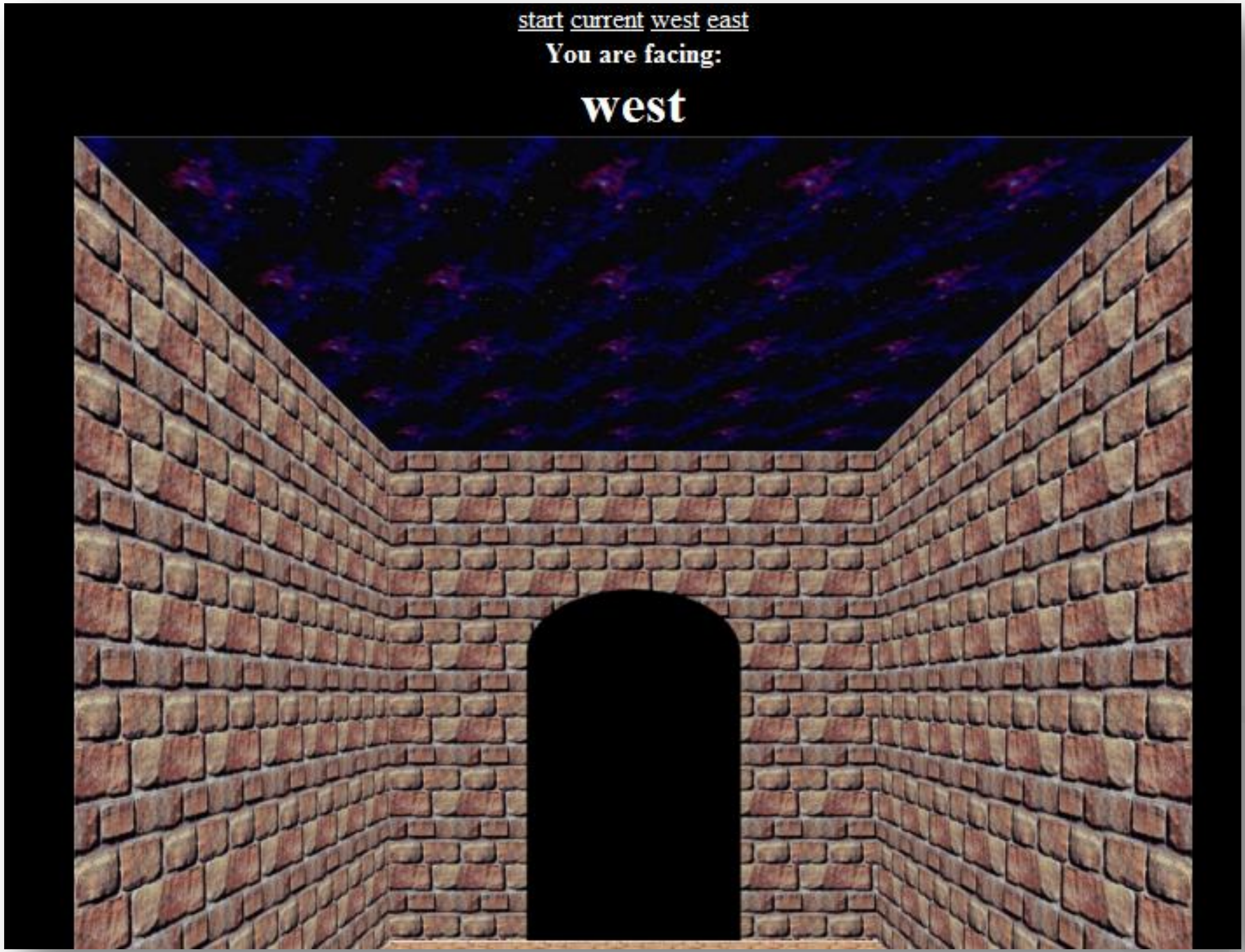
Server



```
private void Get(string[] accepted)
{
  string accept = MimeParse.BestMatch(accepted);
  if(accept==string.Empty)
  {
    Options(406);
    return;
  }

  id = ctx.Request["id"];
  mv = ctx.Request["mv"];
  if (id == string.Empty)
  {
    rtn = GetMazeList(accept);
  }
  else
  {
    if (mv != string.Empty)
    {
      rtn = GetMove(id, mv, accept);
    }
    else
    {
      rtn = GetMaze(id,accept);
    }
  }

  ctx.Response.StatusCode = 200;
  ctx.Response.StatusDescription = "OK";
  ctx.Response.ContentType = accept;
  ctx.Response.Write(rtn);
}
```

Maze+XML

Client

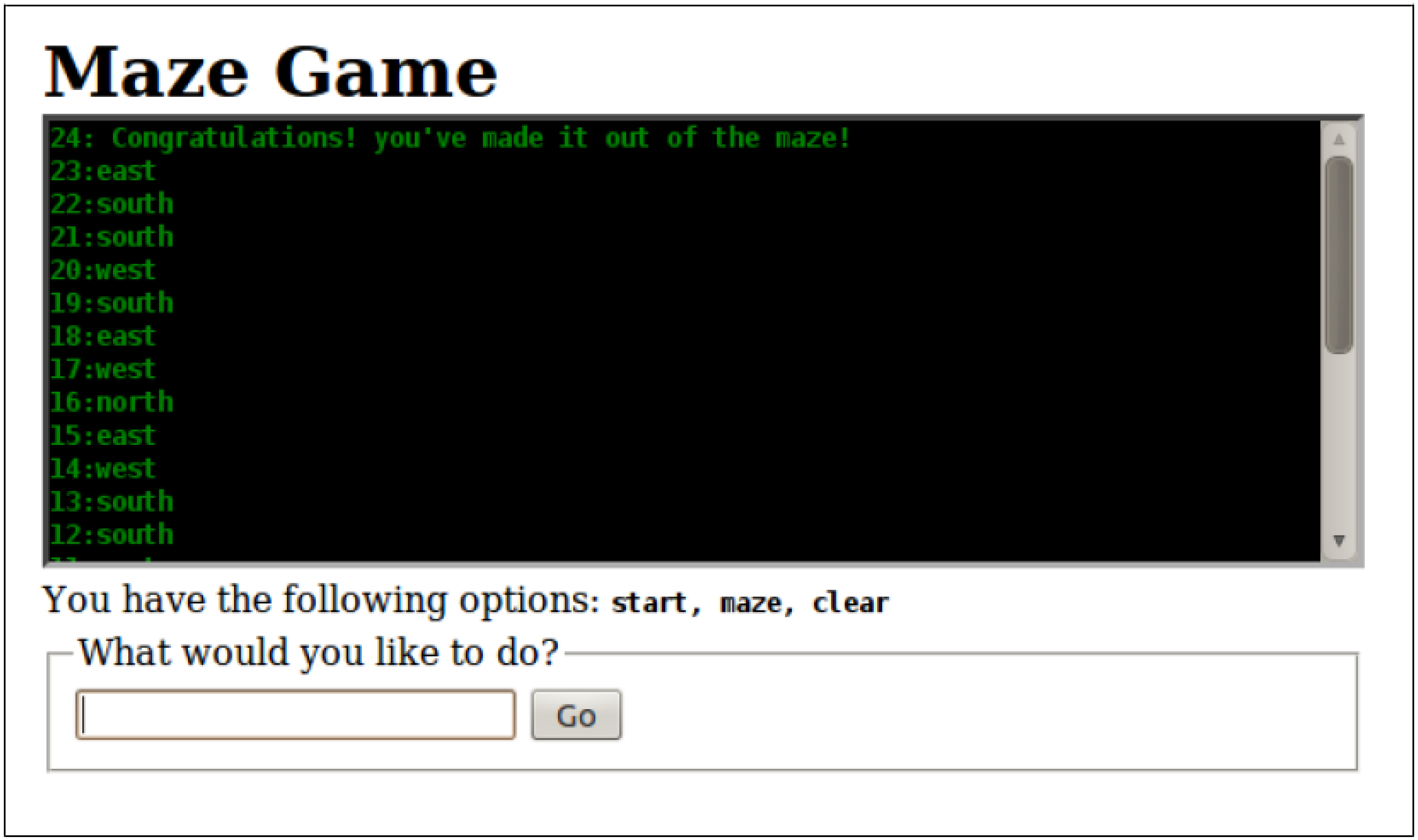

start current west east

You are facing:

west

Maze+XML

Client



Maze Game

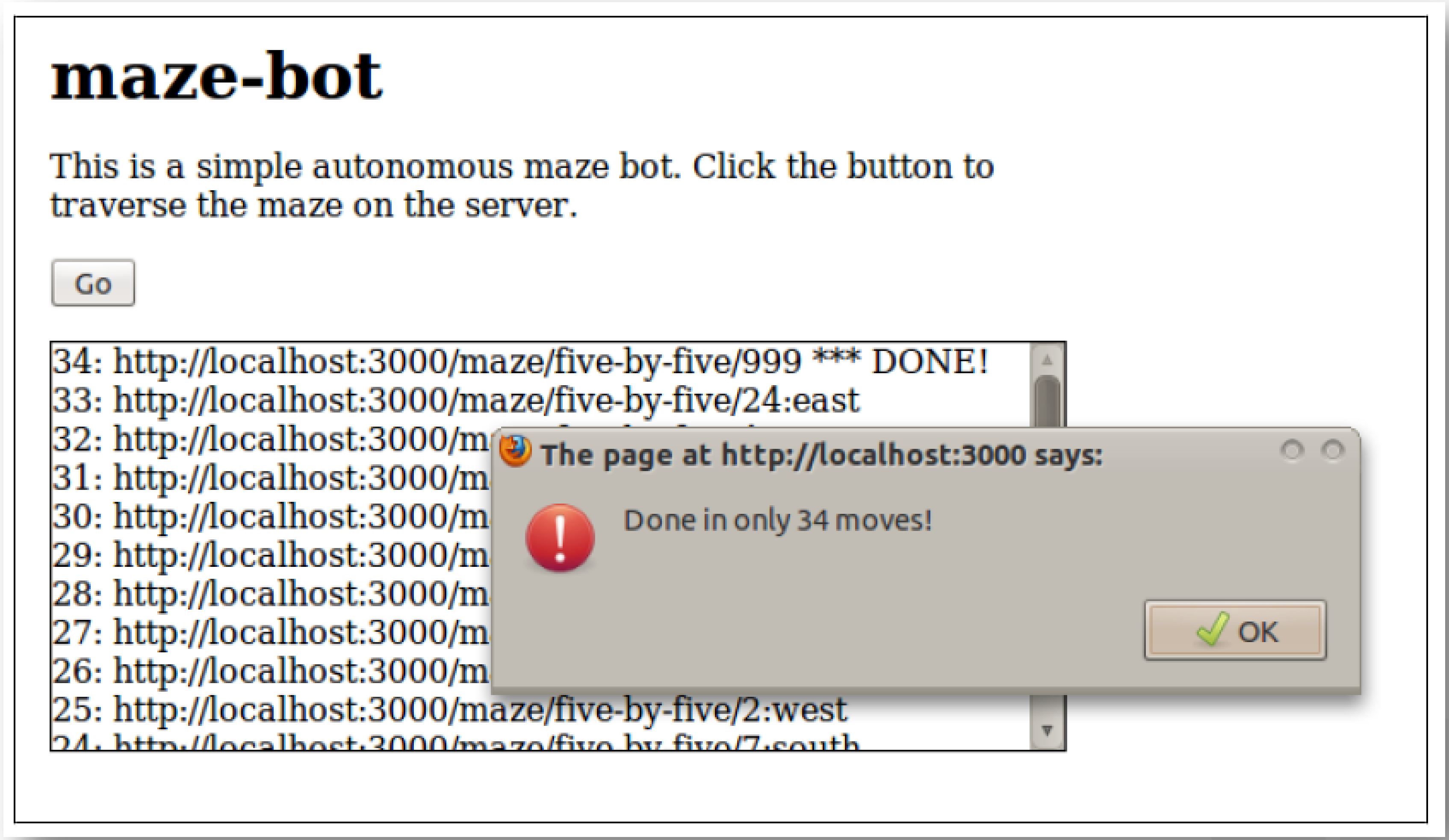
```
24: Congratulations! you've made it out of the maze!
23:east
22:south
21:south
20:west
19:south
18:east
17:west
16:north
15:east
14:west
13:south
12:south
```

You have the following options: start, maze, clear

What would you like to do?

Maze+XML

Client



maze-bot

This is a simple autonomous maze bot. Click the button to traverse the maze on the server.

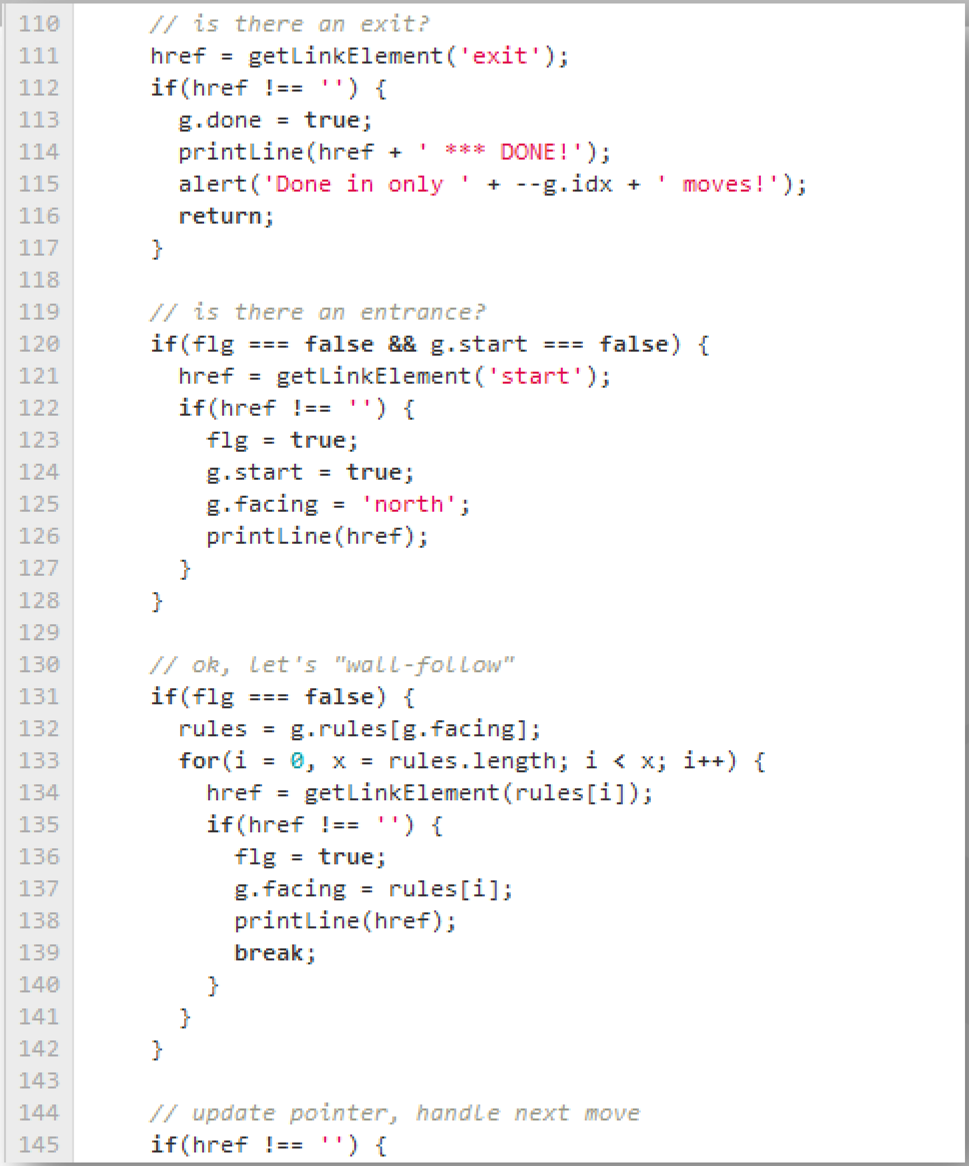
```
34: http://localhost:3000/maze/five-by-five/999 *** DONE!
33: http://localhost:3000/maze/five-by-five/24:east
32: http://localhost:3000/m
31: http://localhost:3000/m
30: http://localhost:3000/m
29: http://localhost:3000/m
28: http://localhost:3000/m
27: http://localhost:3000/m
26: http://localhost:3000/m
25: http://localhost:3000/maze/five-by-five/2:west
24: http://localhost:3000/maze/five-by-five/7:south
```

The page at http://localhost:3000 says:

Done in only 34 moves!

Maze+XML

Client Code



```
    // is there an exit?
    href = getLinkElement('exit');
    if(href !== '') {
      g.done = true;
      printLine(href + ' *** DONE!');
      alert('Done in only ' + --g.idx + ' moves!');
      return;
    }

    // is there an entrance?
    if(flg === false && g.start === false) {
      href = getLinkElement('start');
      if(href !== '') {
        flg = true;
        g.start = true;
        g.facing = 'north';
        printLine(href);
      }
    }

    // ok, let's "wall-follow"
    if(flg === false) {
      rules = g.rules[g.facing];
      for(i = 0, x = rules.length; i < x; i++) {
        href = getLinkElement(rules[i]);
        if(href !== '') {
          flg = true;
          g.facing = rules[i];
          printLine(href);
          break;
        }
      }
    }

    // update pointer, handle next move
    if(href !== '') {
```

Maze+XML

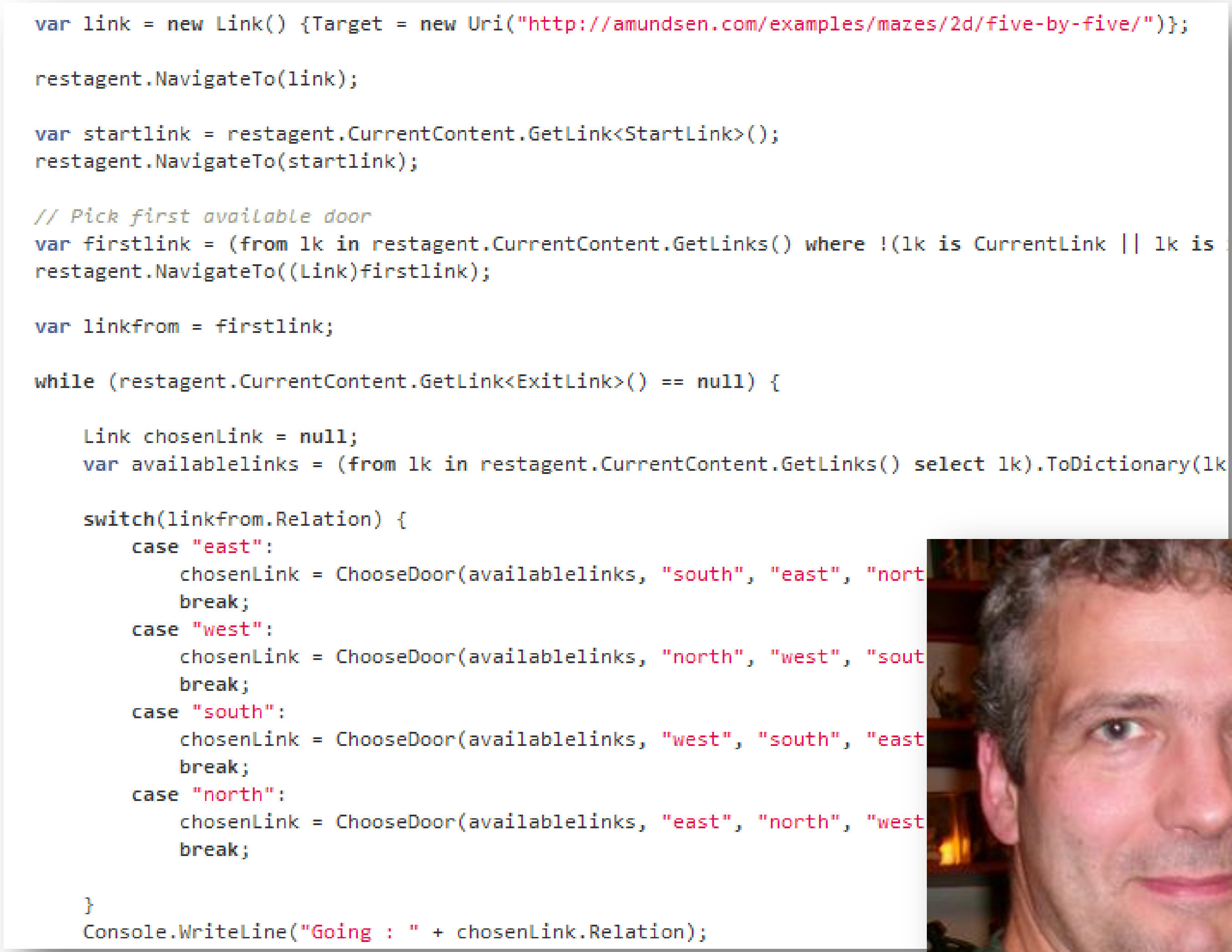
In the wild…

Maze+XML

- **Darrel Miller**
- *"A good example of using link relations to convey domain specific semantics."*
- *"Has been a good test bed for trying to develop a UI transparently that tracks the state of the user agent as it navigates between representations."*

Maze+XML

Darrel Miller – C#



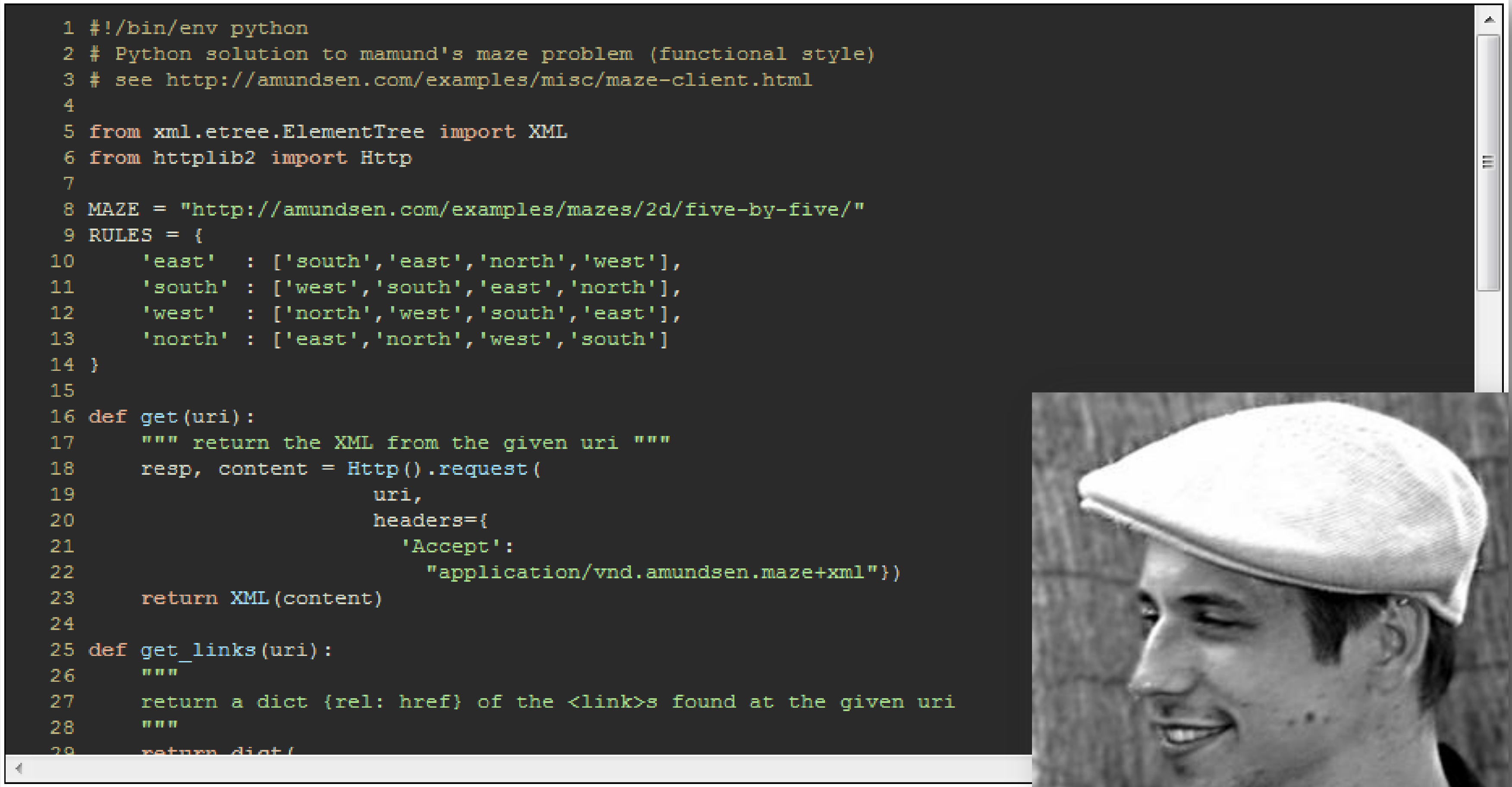
```
var link = new Link() {Target = new Uri("http://amundsen.com/examples/mazes/2d/five-by-five/")};

restagent.NavigateTo(link);

var startlink = restagent.CurrentContent.GetLink<StartLink>();
restagent.NavigateTo(startlink);

// Pick first available door
var firstlink = (from lk in restagent.CurrentContent.GetLinks() where !(lk is CurrentLink || lk is
restagent.NavigateTo((Link)firstlink);

var linkfrom = firstlink;

while (restagent.CurrentContent.GetLink<ExitLink>() == null) {

    Link chosenLink = null;
    var availablelinks = (from lk in restagent.CurrentContent.GetLinks() select lk).ToDictionary(lk

    switch(linkfrom.Relation) {
        case "east":
            chosenLink = ChooseDoor(availablelinks, "south", "east", "nort
            break;
        case "west":
            chosenLink = ChooseDoor(availablelinks, "north", "west", "sout
            break;
        case "south":
            chosenLink = ChooseDoor(availablelinks, "west", "south", "east
            break;
        case "north":
            chosenLink = ChooseDoor(availablelinks, "east", "north", "west
            break;

    }
    Console.WriteLine("Going : " + chosenLink.Relation);
```

Maze+XML

- **Yannick Loiseau**
- *"I can say that a non-restful architecture would have been a lot harder to deal with in bash, because hypermedia obviously made the maze exploration really easy"*
- *"I think that Link headers would be even easier to deal with…"*

Maze+XML

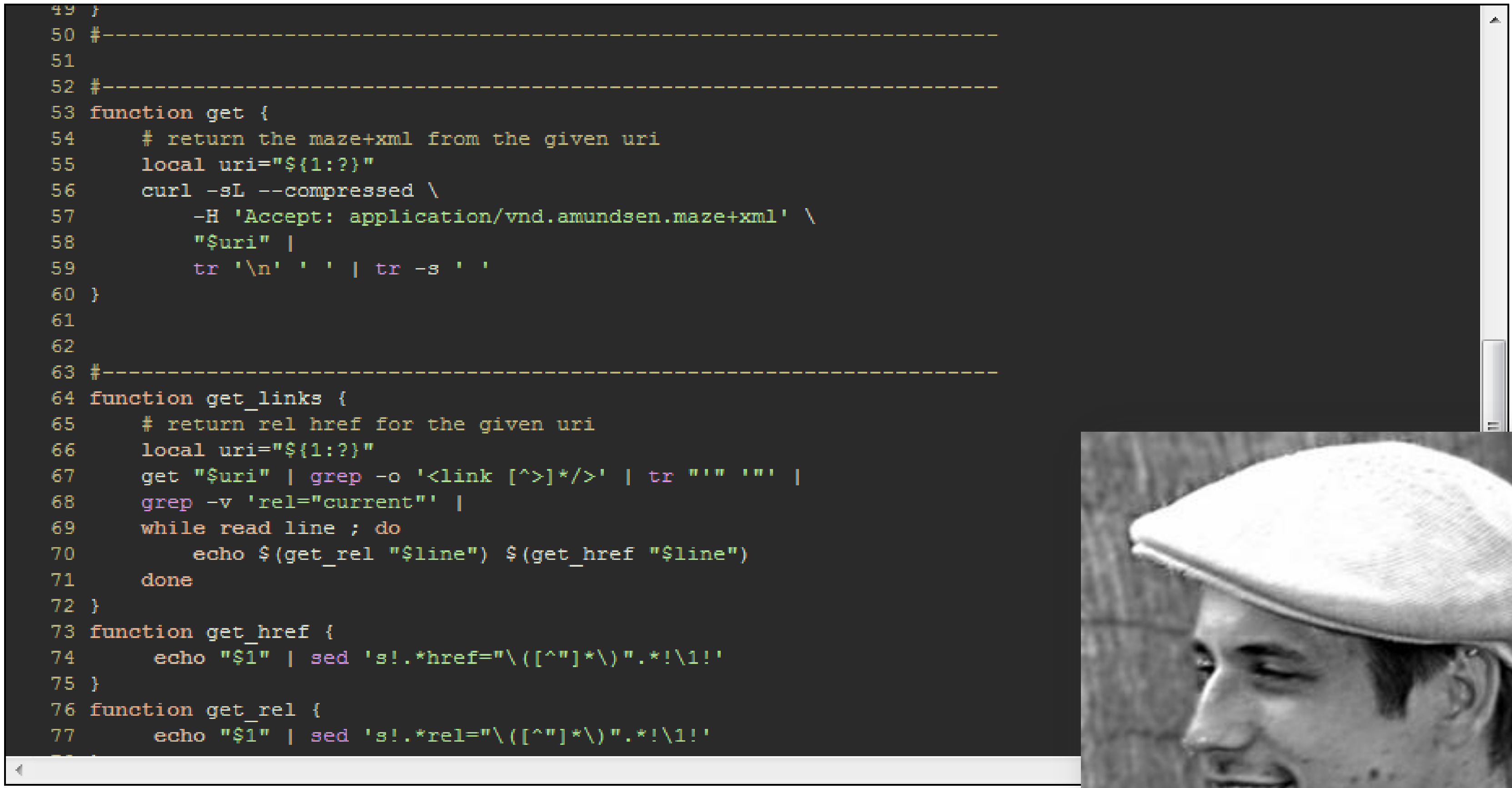
Yannick Loiseau - Python



```
#!/bin/env python
# Python solution to mamund's maze problem (functional style)
# see http://amundsen.com/examples/misc/maze-client.html

from xml.etree.ElementTree import XML
from httplib2 import Http

MAZE = "http://amundsen.com/examples/mazes/2d/five-by-five/"
RULES = {
    'east'  : ['south','east','north','west'],
    'south' : ['west','south','east','north'],
    'west'  : ['north','west','south','east'],
    'north' : ['east','north','west','south']
}

def get(uri):
    """ return the XML from the given uri """
    resp, content = Http().request(
                        uri,
                        headers={
                          'Accept':
                            "application/vnd.amundsen.maze+xml"})
    return XML(content)

def get_links(uri):
    """
    return a dict {rel: href} of the <link>s found at the given uri
    """
```

Maze+XML

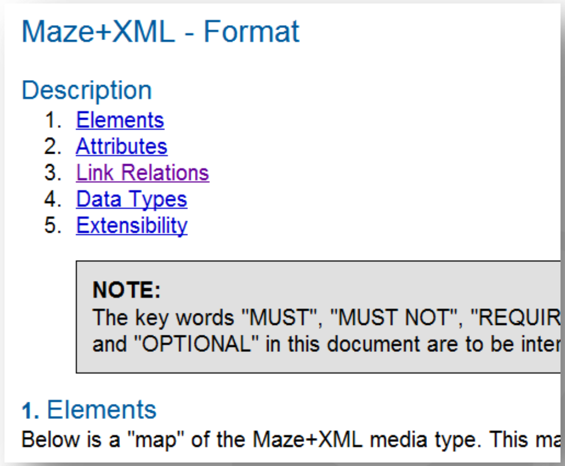
Yannick Loiseau - Bash



```
}
#----------------------------------------------------------------------

#----------------------------------------------------------------------
function get {
    # return the maze+xml from the given uri
    local uri="${1:?}"
    curl -sL --compressed \
        -H 'Accept: application/vnd.amundsen.maze+xml' \
        "$uri" |
        tr '\n' ' ' | tr -s ' '
}


#----------------------------------------------------------------------
function get_links {
    # return rel href for the given uri
    local uri="${1:?}"
    get "$uri" | grep -o '<link [^>]*/>' | tr "'" '"' |
    grep -v 'rel="current"' |
    while read line ; do
        echo $(get_rel "$line") $(get_href "$line")
    done
}
function get_href {
     echo "$1" | sed 's!.*href="\([^"]*\)".*!\1!'
}
function get_rel {
     echo "$1" | sed 's!.*rel="\([^"]*\)".*!\1!'
```

Maze+XML

- Characteristics
  - Read-Only navigational links
  - Limited set of identifiers
  - Domain specific
- Benefits
  - Simple, direct design
  - Easy to create servers/clients
  - M2M works when algorithm is available
- Costs
  - Limited reach
  - M2M clients challenge evolvability



Maze+XML - Format

Description

1. Elements
2. Attributes
3. Link Relations
4. Data Types
5. Extensibility

NOTE:
The key words "MUST", "MUST NOT", "REQUIR
and "OPTIONAL" in this document are to be inter

1. Elements

Below is a "map" of the Maze+XML media type. This ma

H-Factors

“Pardon me, did you say ‘links’?”

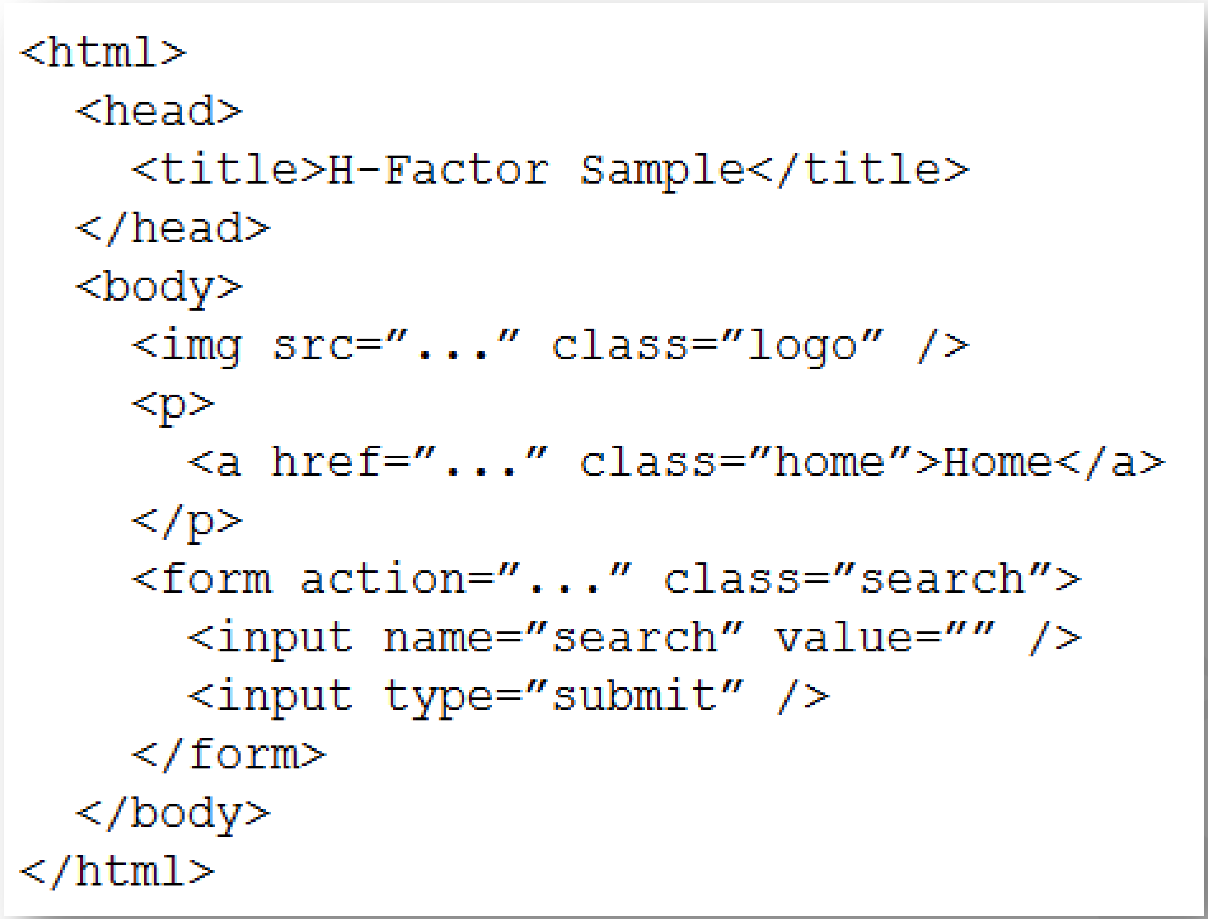
H-Factors

- *“The **H Factor** of a media-type is a measure of the level of hypermedia support within that media-type.”*
- *“**H Factor** values can be used to compare and contrast media types in order to aid in selecting the proper media-type(s) for your implementation.”*

REST: From Research to Practice : Hypermedia Types, 2011
- Mike Amundsen

H-Factors

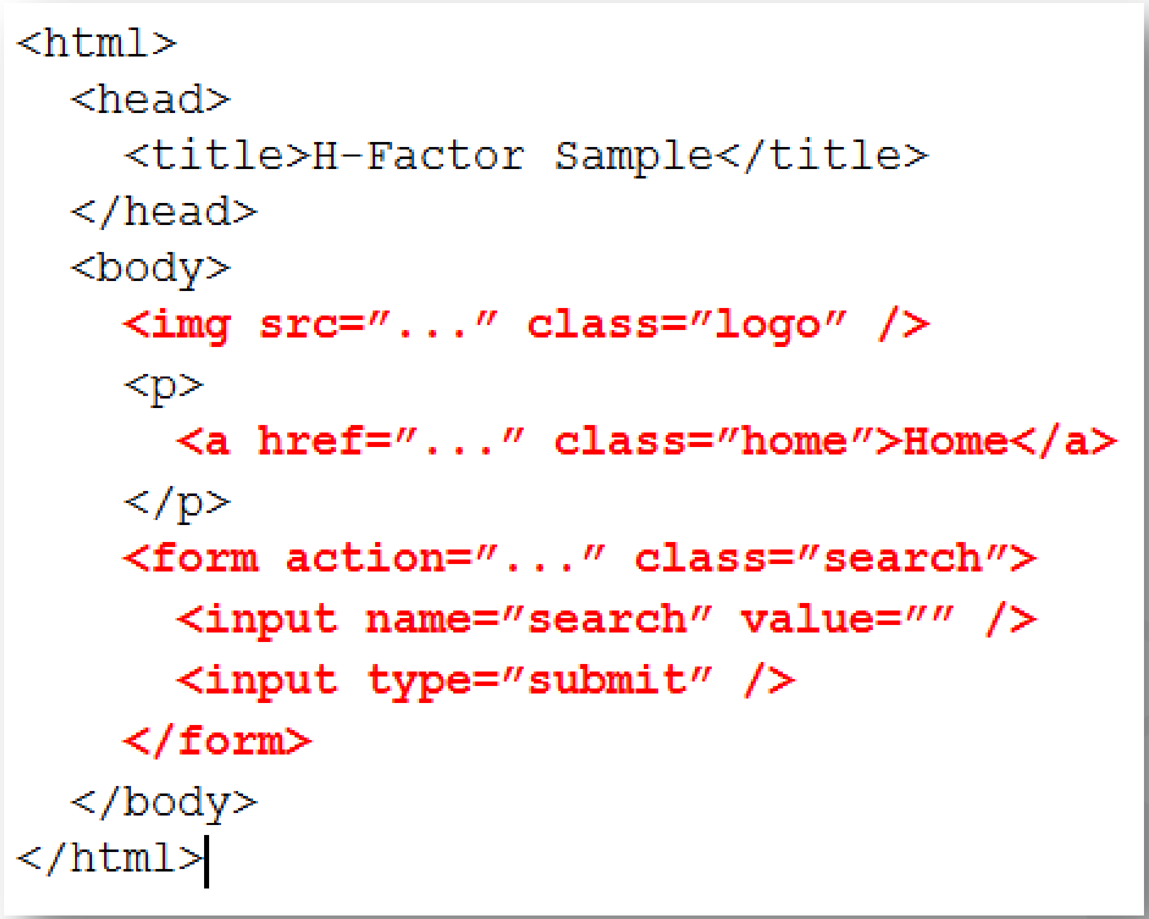
Analyzing Media Types



```
<html>
  <head>
    <title>H-Factor Sample</title>
  </head>
  <body>
    <img src="..." class="logo" />
    <p>
      <a href="..." class="home">Home</a>
    </p>
    <form action="..." class="search">
      <input name="search" value="" />
      <input type="submit" />
    </form>
  </body>
</html>
```

H-Factors

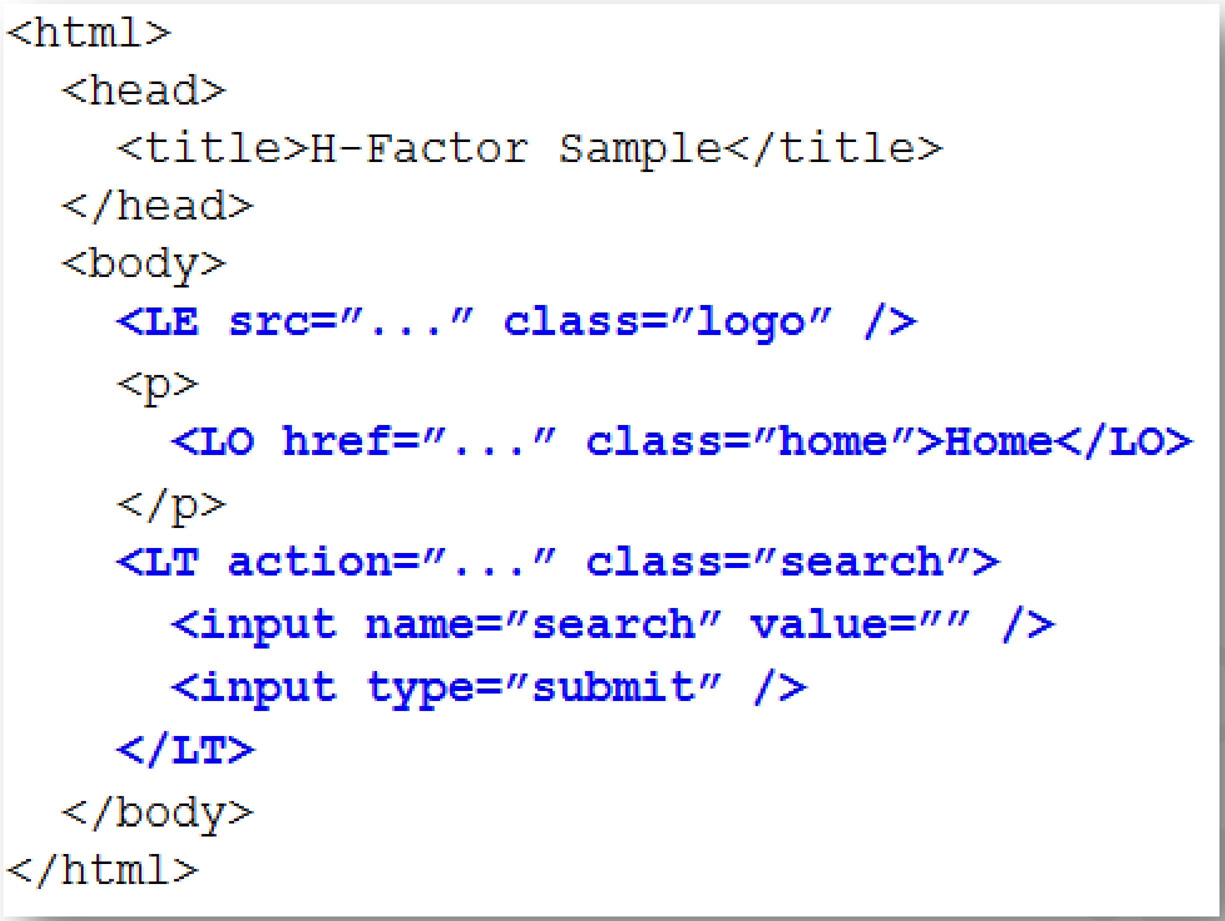
Analyzing Media Types



```
<html>
  <head>
    <title>H-Factor Sample</title>
  </head>
  <body>
    <img src="..." class="logo" />
    <p>
      <a href="..." class="home">Home</a>
    </p>
    <form action="..." class="search">
      <input name="search" value="" />
      <input type="submit" />
    </form>
  </body>
</html>
```

H-Factors

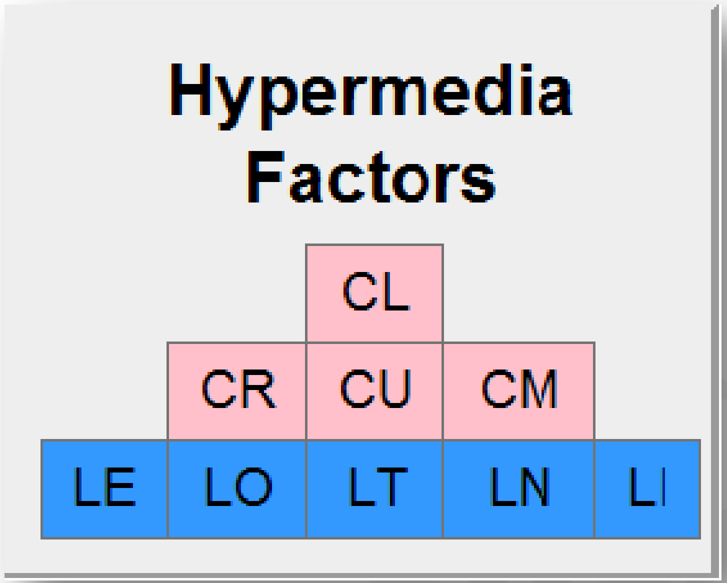
Analyzing Media Types



```
<html>
  <head>
    <title>H-Factor Sample</title>
  </head>
  <body>
    <LE src="..." class="logo" />
    <p>
      <LO href="..." class="home">Home</LO>
    </p>
    <LT action="..." class="search">
      <input name="search" value="" />
      <input type="submit" />
    </LT>
  </body>
</html>
```

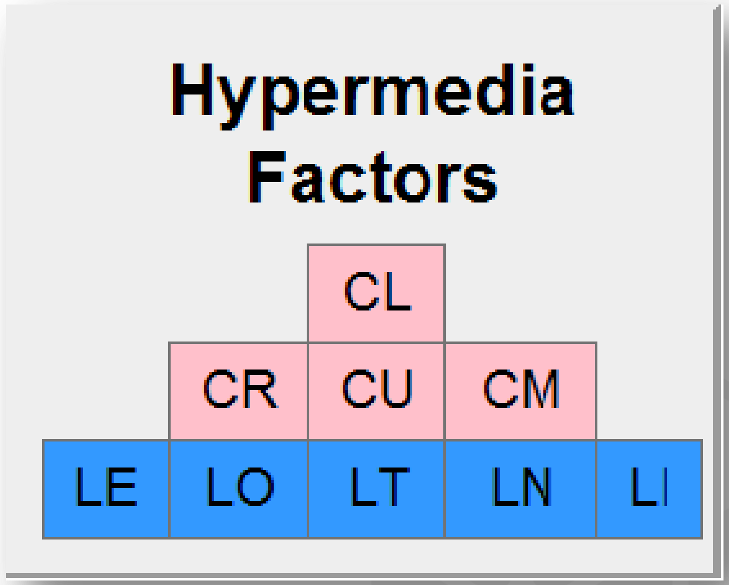
H-Factors

- There are five LINK Factors (LO, LE, LT, LI, LN)
- There are four CONTROL Factors (CR, CU, CM, CL)



H-Factors

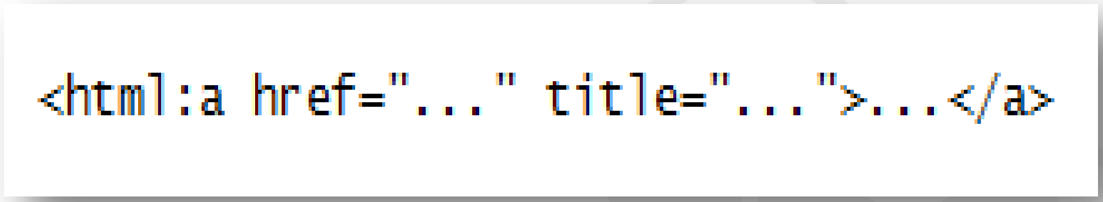
- **There are five LINK Factors (LO, LE, LT, LI, LN)**
- There are four CONTROL Factors (CR, CU, CM, CL)



H-Factors

Linking

- **Outbound Links (LO)**

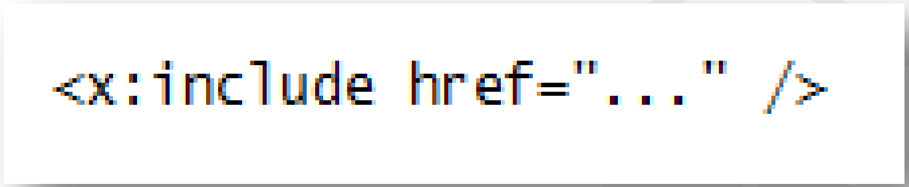


```
<html:a href="..." title="...">...</a>
```

H-Factors

Linking

- **Outbound Links (LO)**
- **Embedded Links (LE)**

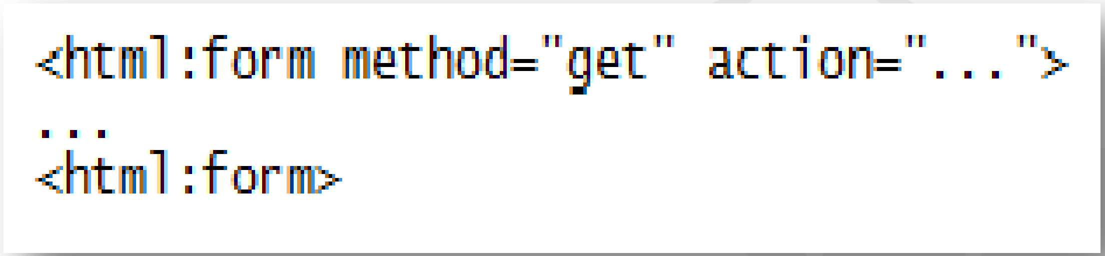


```
<x:include href="..." />
```

H-Factors

Linking

- **Outbound Links (LO)**
- **Embedded Links (LE)**
- **Templated Links (LT)**

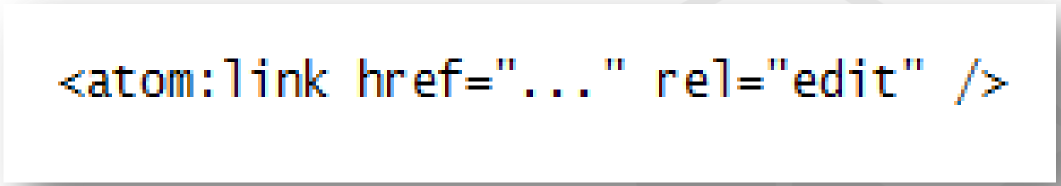


```
<html:form method="get" action="...">
...
<html:form>
```

H-Factors

Linking

- **Outbound Links (LO)**
- **Embedded Links (LE)**
- **Templated Links (LT)**
- **Idempotent Links (LI)**

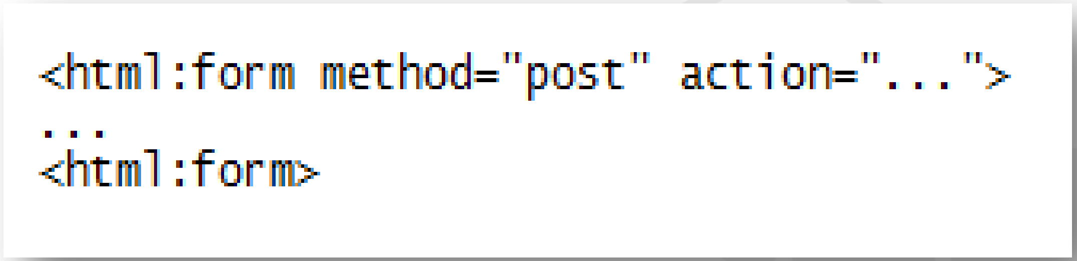


```
<atom:link href="..." rel="edit" />
```

H-Factors

Linking

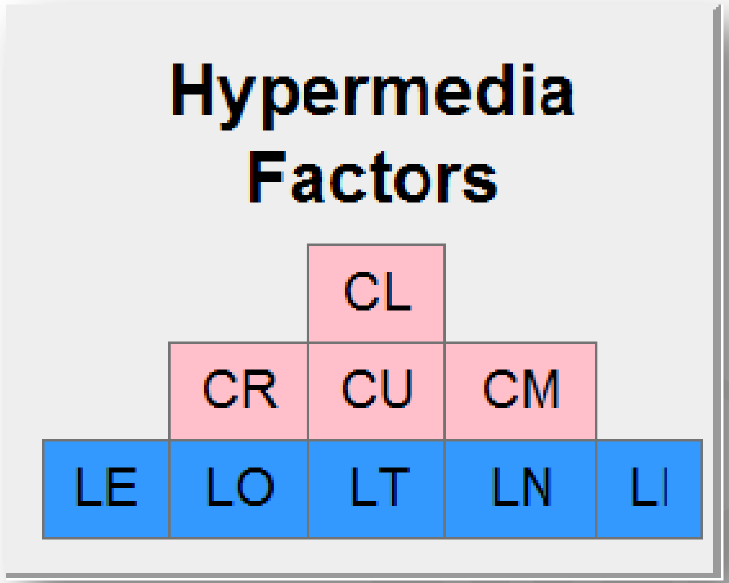
- Outbound Links (LO)
- Embedded Links (LE)
- Templated Links (LT)
- Idempotent Links (LI)
- Non-Idempotent Links (LN)



```
<html:form method="post" action="...">
...
<html:form>
```

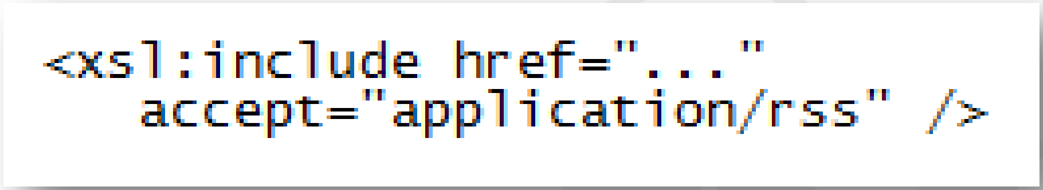
H-Factors

- There are five LINK Factors (LO, LE, LT, LI, LN)
- **There are four CONTROL Factors (CR, CU, CM, CL)**



Control

- **Request Controls (CR)**

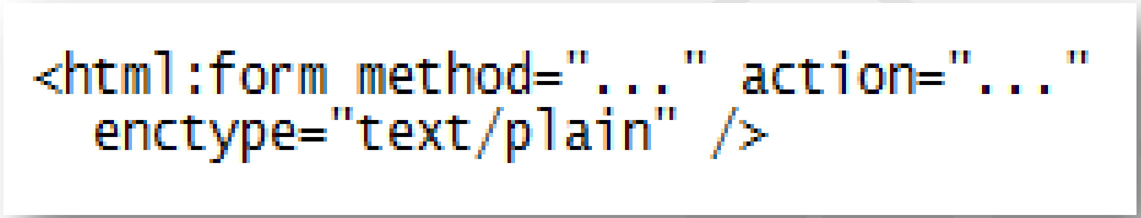


```
<xsl:include href="..."
    accept="application/rss" />
```

H-Factors

Control

- Request Controls (CR)
- Update Controls (CU)

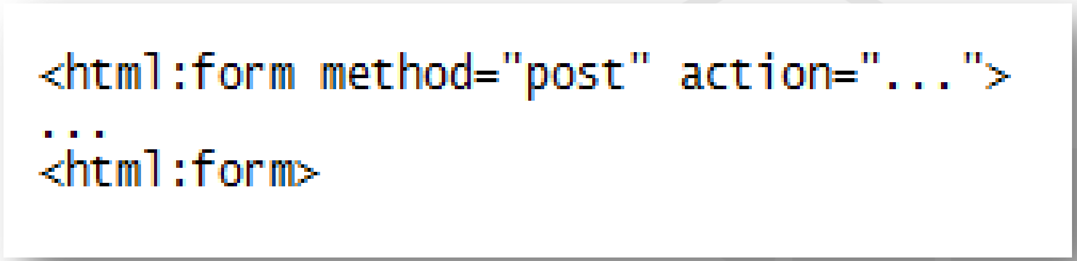


```
<html:form method="..." action="..."
  enctype="text/plain" />
```

H-Factors

Control

- Request Controls (CR)
- Update Controls (CU)
- Method Controls (CM)

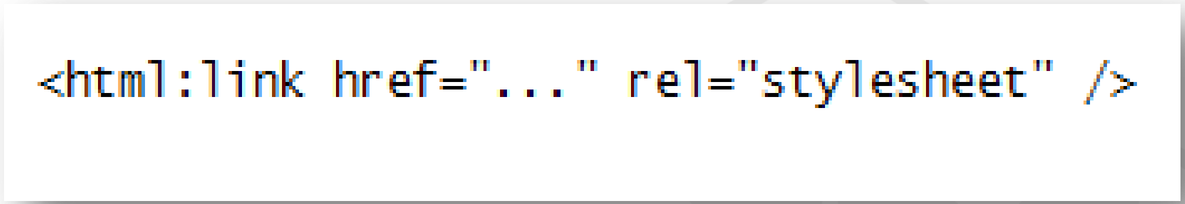


```
<html:form method="post" action="...">
...
<html:form>
```

H-Factors

Control

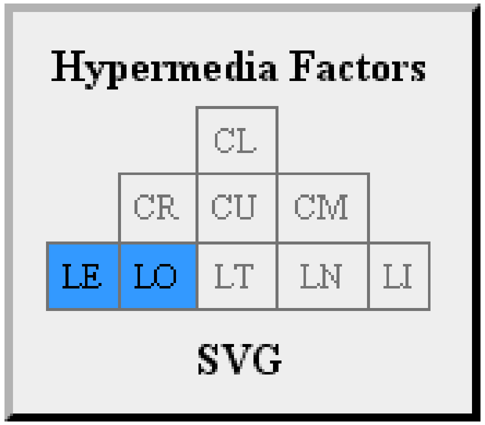
- Request Controls (CR)
- Update Controls (CU)
- Method Controls (CM)
- Link Controls (CL)

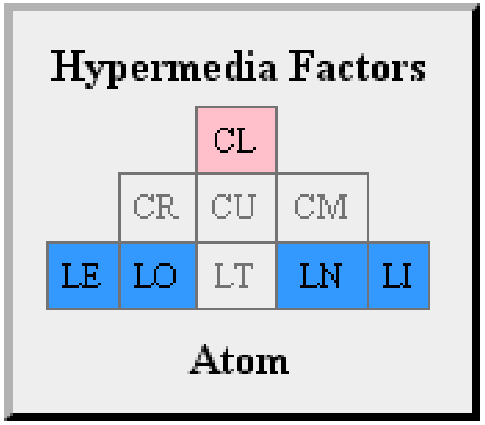


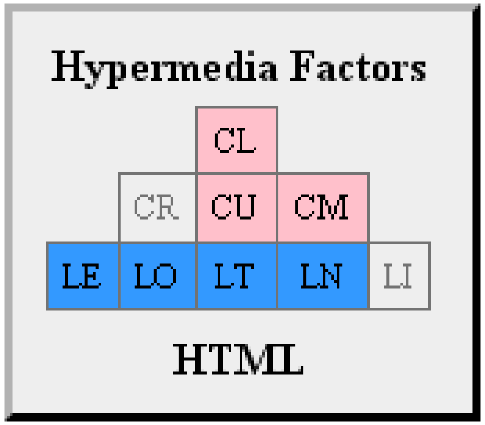
```
<html:link href="..." rel="stylesheet" />
```

H-Factors

- A pre-defined collection of H-Factors is called a “Media Type”
- Each media type has it’s own “H-Factor” signature.







H-Factors

H-Factors document the Affordances of the Media Type

H-Factors

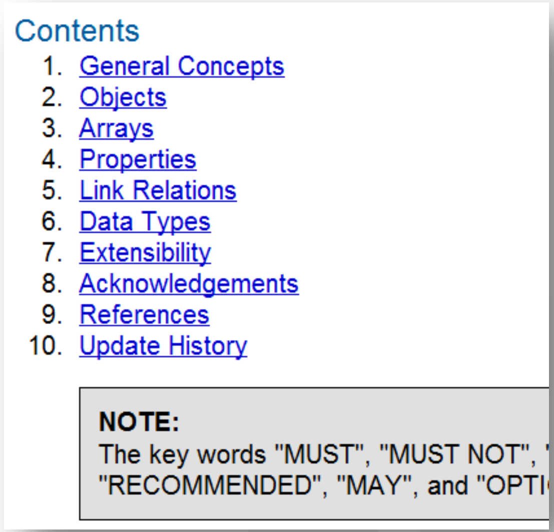
“OK, media types, affordances, I see…

Collection+JSON

Design #2:
A read/write hypermedia type

Collection+JSON

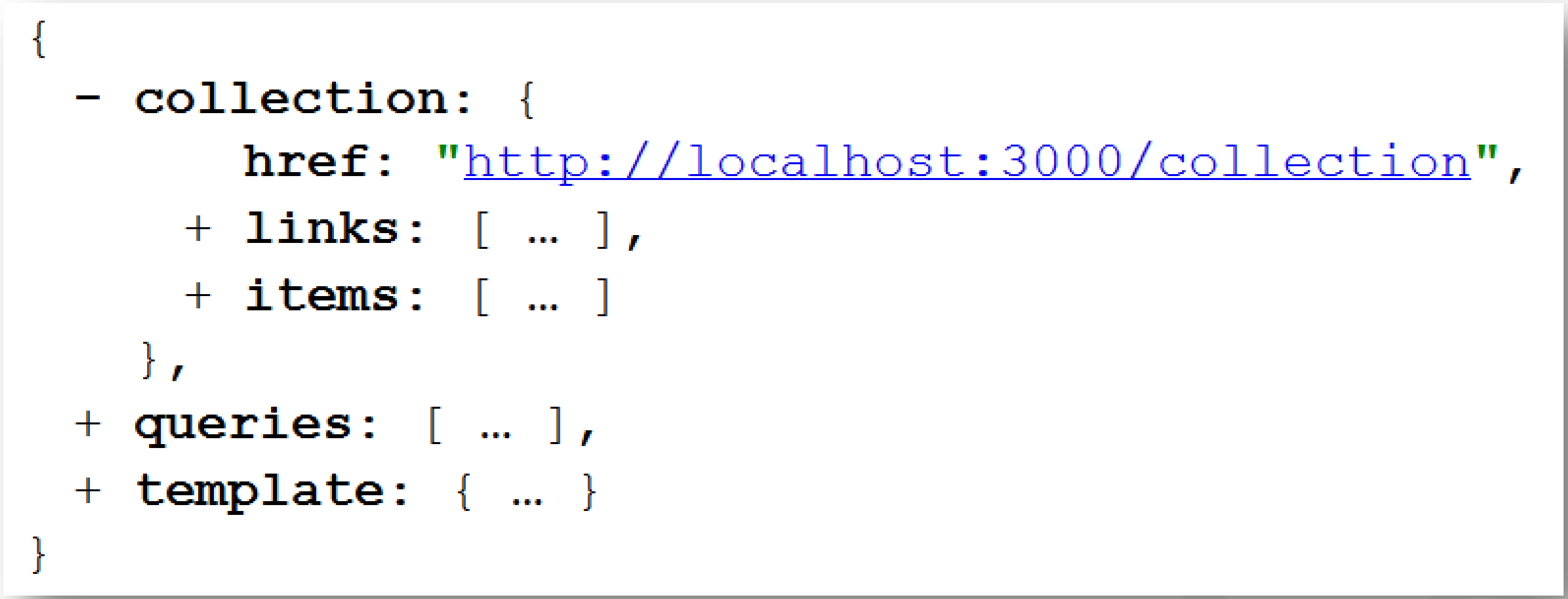
- **Collection+JSON** media type
- First designs in early 2011, registered w/ IANA mid 2011
- *“…a JSON-based read/write hypermedia-type designed to support management and querying of simple collections.”*
- It’s Atom w/ LT + templated writes
- Very limited link identifiers
  - collection, item, templates, query


Contents

NOTE:
The key words "MUST", "MUST NOT", "
"RECOMMENDED", "MAY", and "OPTI

Collection+JSON

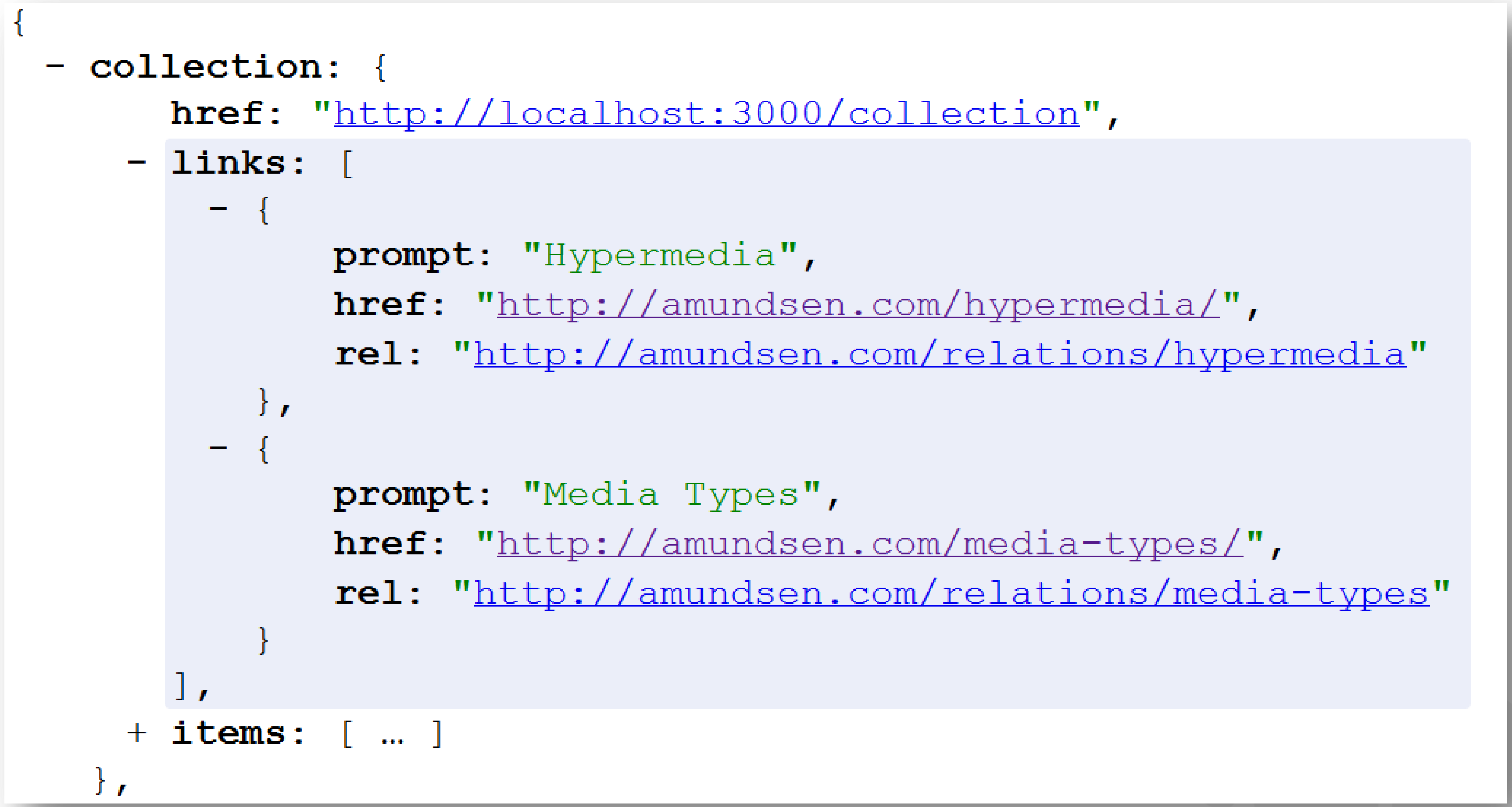
Message



```
{
  - collection: {
        href: "http://localhost:3000/collection",
      + links: [ … ],
      + items: [ … ]
    },
  + queries: [ … ],
  + template: { … }
}
```

Collection+JSON

Message

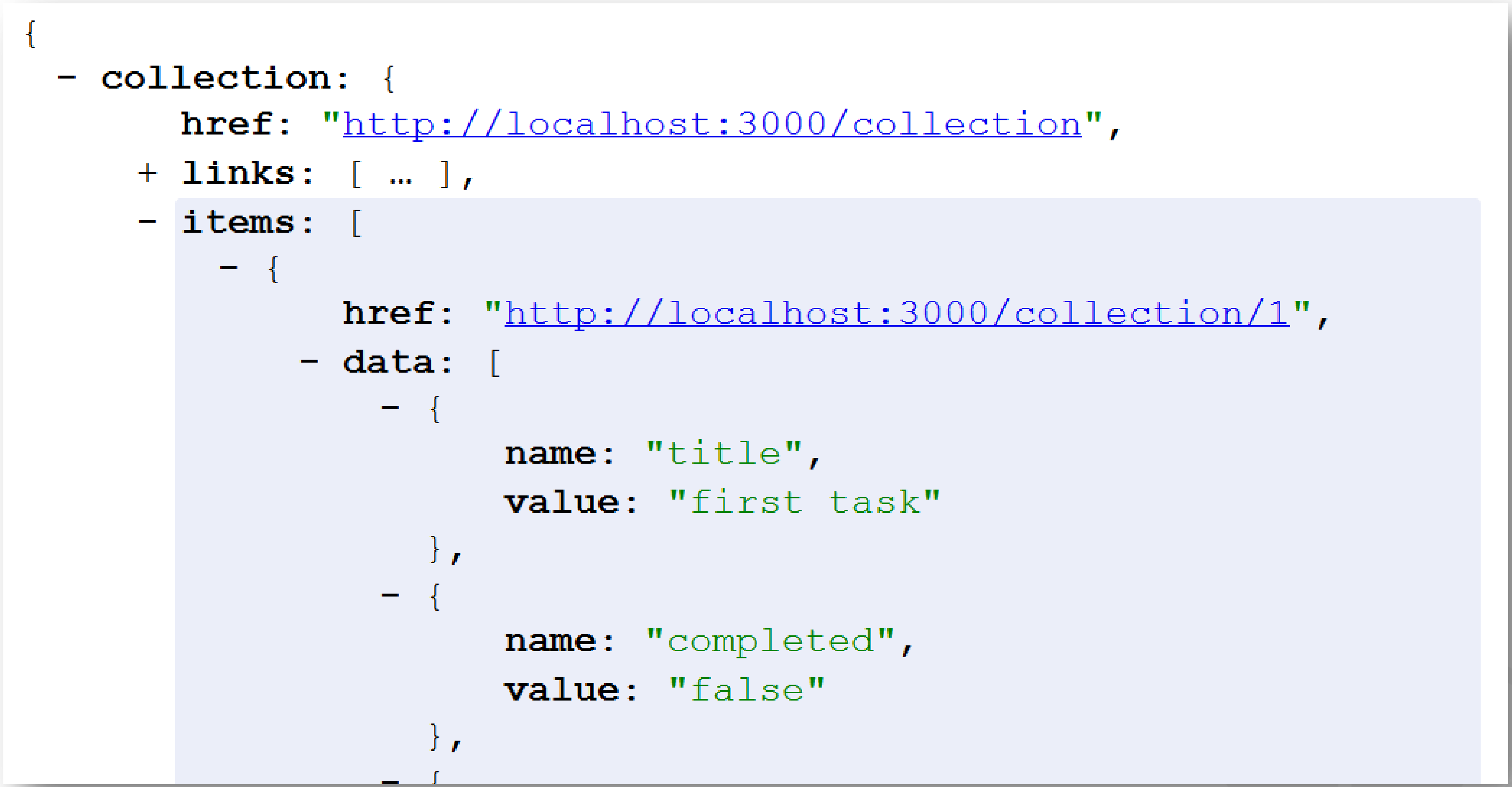


```
{
  - collection: {
        href: "http://localhost:3000/collection",
      - links: [
          - {
                prompt: "Hypermedia",
                href: "http://amundsen.com/hypermedia/",
                rel: "http://amundsen.com/relations/hypermedia"
            },
          - {
                prompt: "Media Types",
                href: "http://amundsen.com/media-types/",
                rel: "http://amundsen.com/relations/media-types"
            }
        ],
      + items: [ … ]
    },
```

LAYER 7 TECHNOLOGIES

Collection+JSON

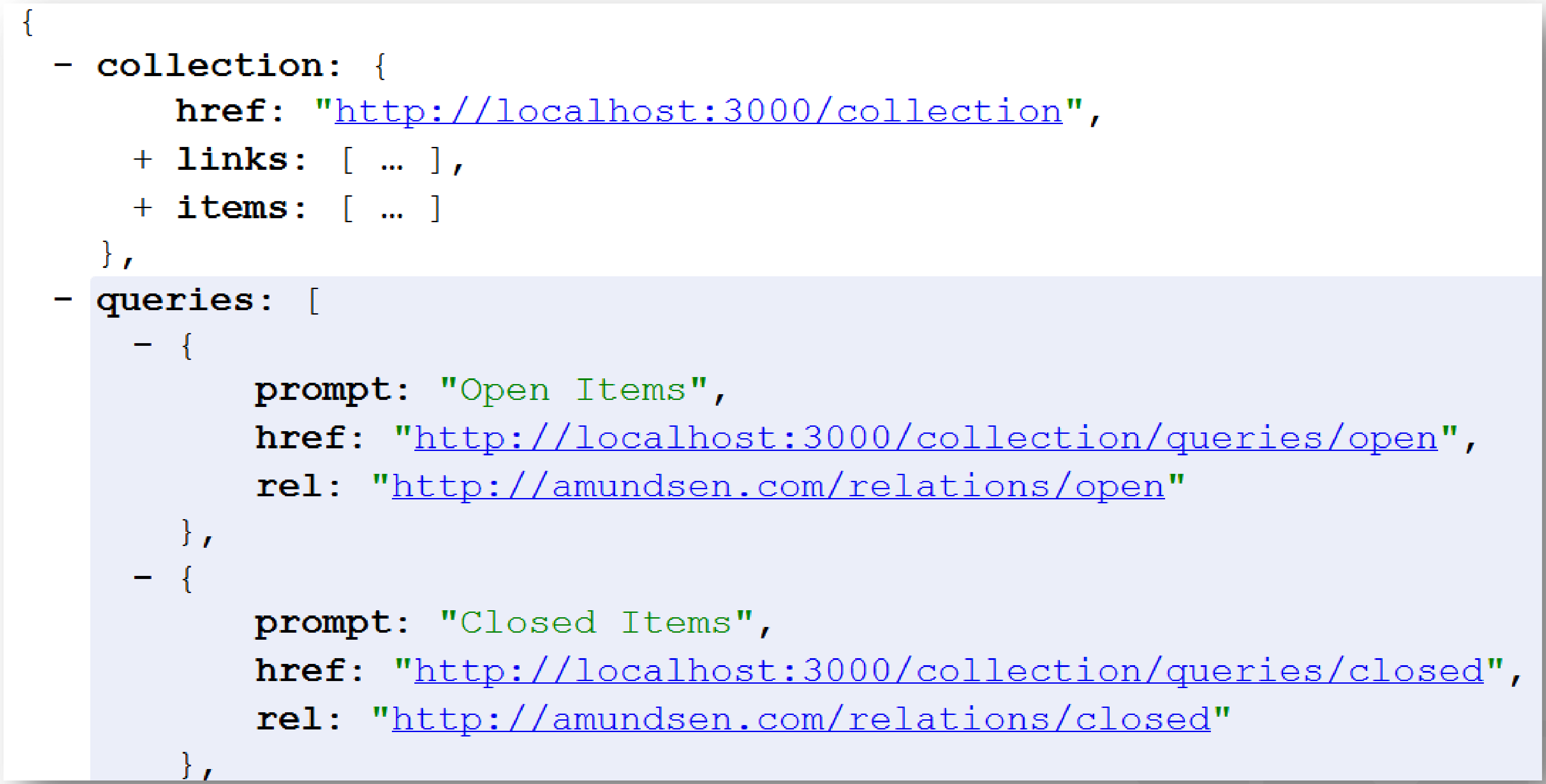
Message



```
{
  - collection: {
        href: "http://localhost:3000/collection",
      + links: [ … ],
      - items: [
          - {
                href: "http://localhost:3000/collection/1",
              - data: [
                  - {
                        name: "title",
                        value: "first task"
                    },
                  - {
                        name: "completed",
                        value: "false"
                    },
```

Collection+JSON

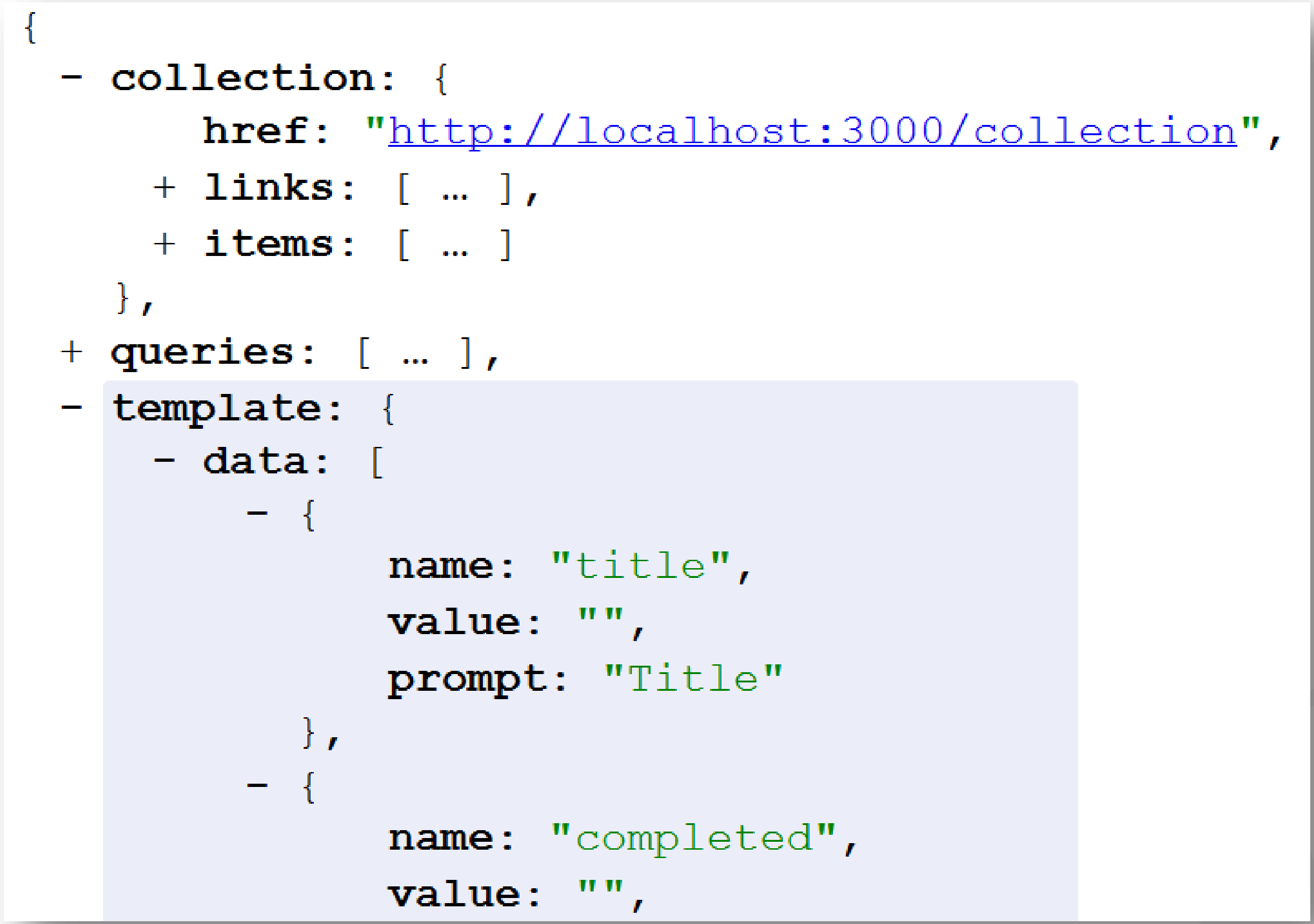
Message



```
{
 - collection: {
       href: "http://localhost:3000/collection",
     + links: [ … ],
     + items: [ … ]
   },
 - queries: [
     - {
           prompt: "Open Items",
           href: "http://localhost:3000/collection/queries/open",
           rel: "http://amundsen.com/relations/open"
       },
     - {
           prompt: "Closed Items",
           href: "http://localhost:3000/collection/queries/closed",
           rel: "http://amundsen.com/relations/closed"
       },
```

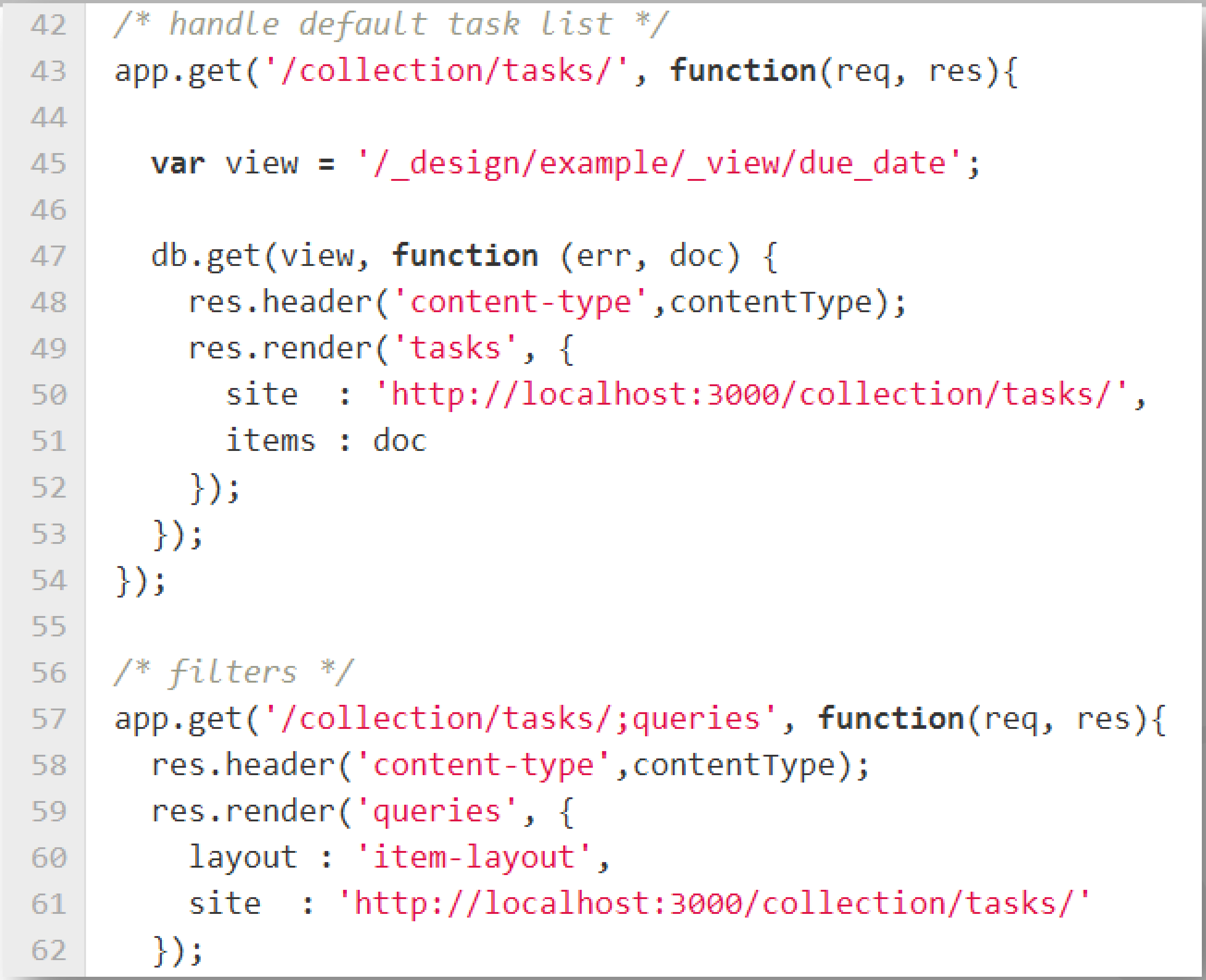
Collection+JSON

Message


```
{
  - collection: {
        href: "http://localhost:3000/collection",
      + links: [ … ],
      + items: [ … ]
    },
  + queries: [ … ],
  - template: {
      - data: [
          - {
                name: "title",
                value: "",
                prompt: "Title"
            },
          - {
                name: "completed",
                value: "",
```

Collection+JSON

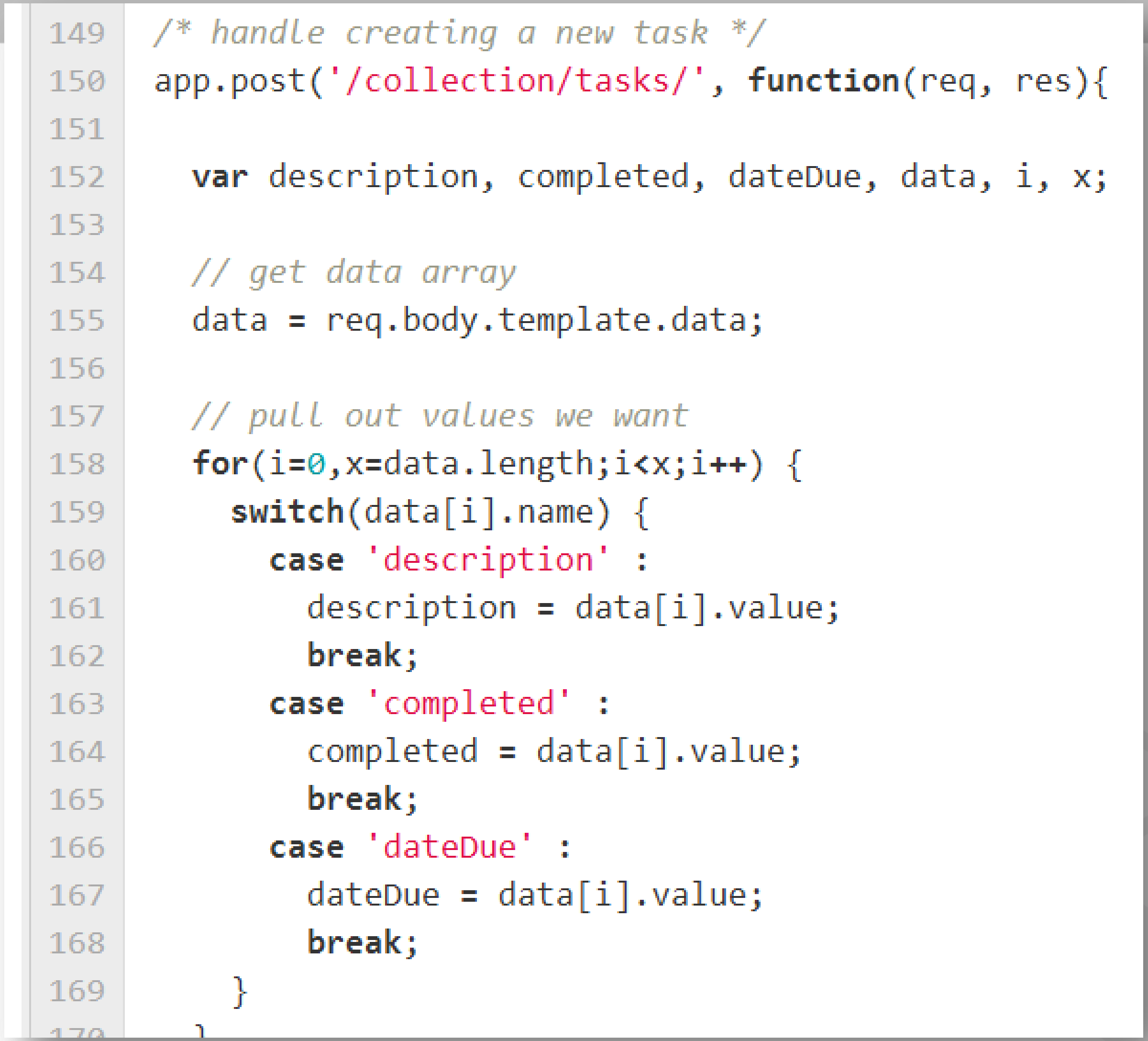
Server



```
/* handle default task list */
app.get('/collection/tasks/', function(req, res){

  var view = '/_design/example/_view/due_date';

  db.get(view, function (err, doc) {
    res.header('content-type',contentType);
    res.render('tasks', {
      site  : 'http://localhost:3000/collection/tasks/',
      items : doc
    });
  });
});

/* filters */
app.get('/collection/tasks/;queries', function(req, res){
  res.header('content-type',contentType);
  res.render('queries', {
    layout : 'item-layout',
    site   : 'http://localhost:3000/collection/tasks/'
  });
```

Collection+JSON

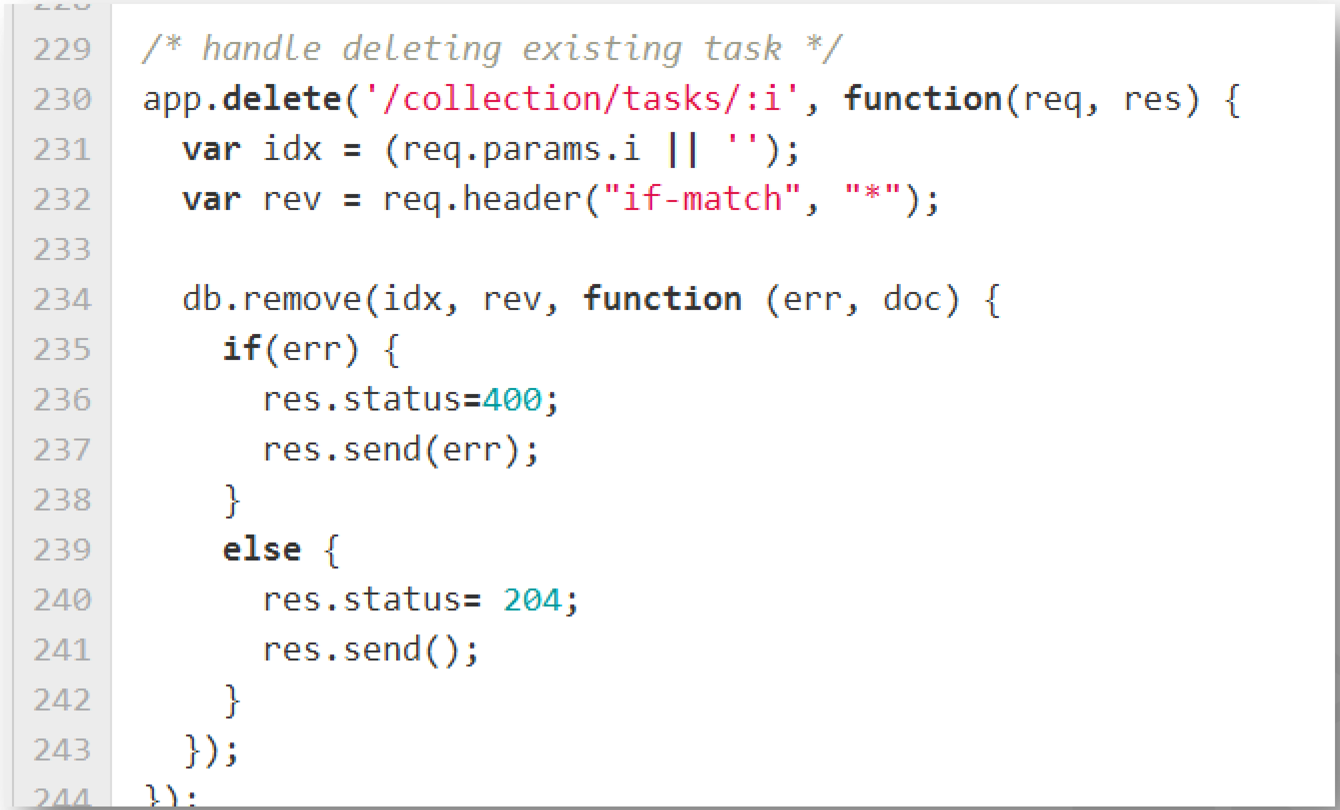
Server



```
/* handle creating a new task */
app.post('/collection/tasks/', function(req, res){

  var description, completed, dateDue, data, i, x;

  // get data array
  data = req.body.template.data;

  // pull out values we want
  for(i=0,x=data.length;i<x;i++) {
    switch(data[i].name) {
      case 'description' :
        description = data[i].value;
        break;
      case 'completed' :
        completed = data[i].value;
        break;
      case 'dateDue' :
        dateDue = data[i].value;
        break;
    }
```

Collection+JSON

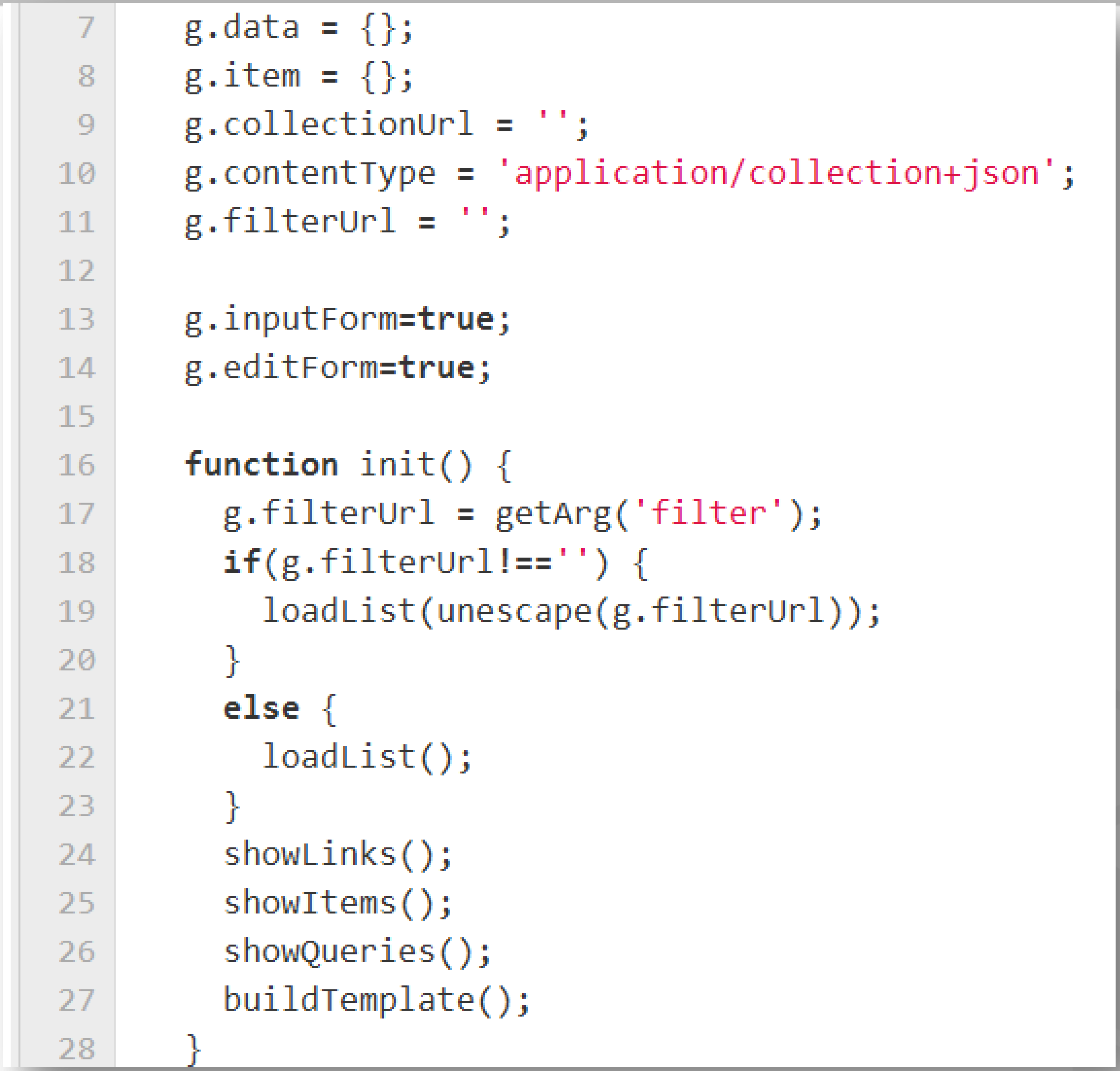
Server



```
/* handle deleting existing task */
app.delete('/collection/tasks/:i', function(req, res) {
  var idx = (req.params.i || '');
  var rev = req.header("if-match", "*");

  db.remove(idx, rev, function (err, doc) {
    if(err) {
      res.status=400;
      res.send(err);
    }
    else {
      res.status= 204;
      res.send();
    }
  });
});
```

Collection+JSON

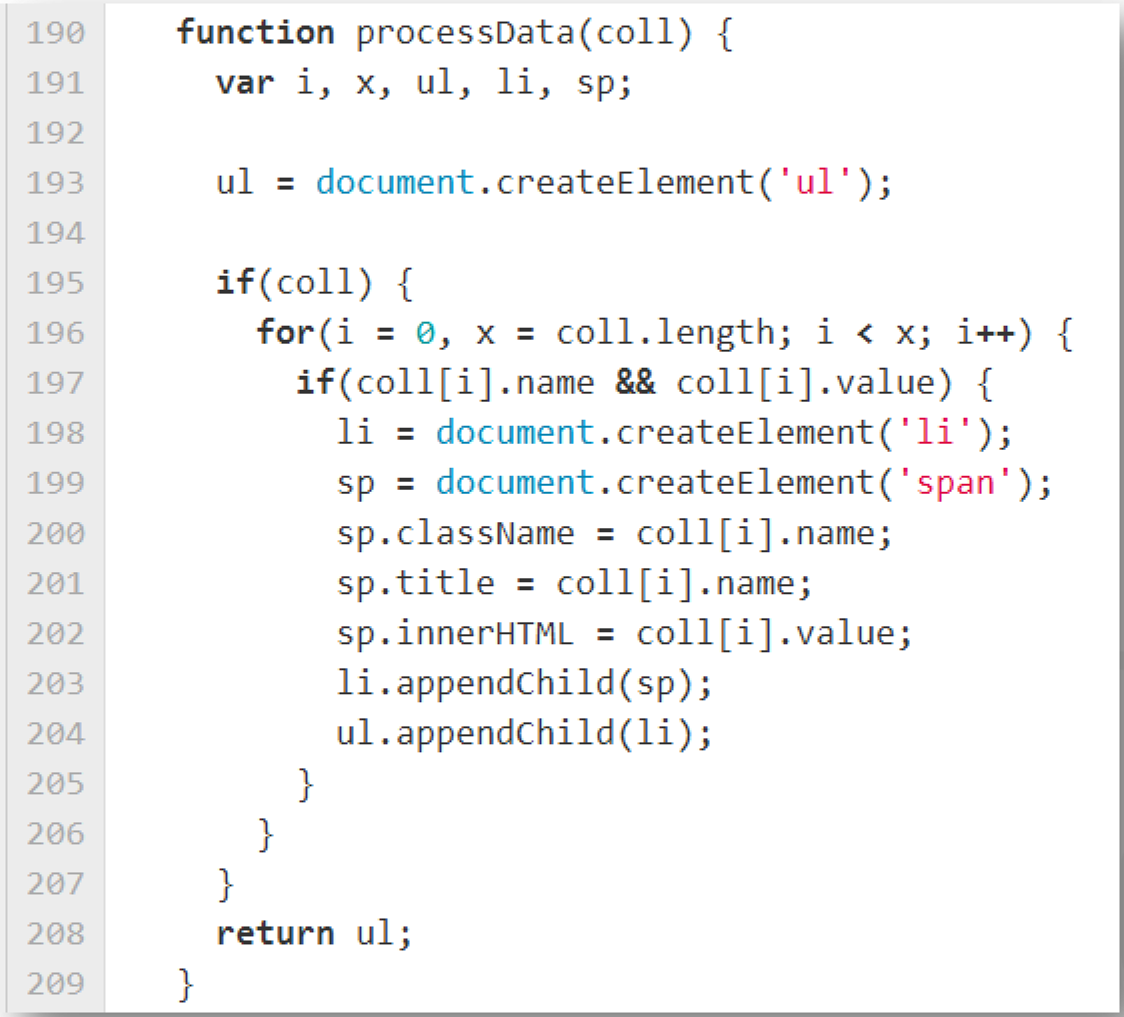
Client Code



```
g.data = {};
g.item = {};
g.collectionUrl = '';
g.contentType = 'application/collection+json';
g.filterUrl = '';

g.inputForm=true;
g.editForm=true;

function init() {
  g.filterUrl = getArg('filter');
  if(g.filterUrl!=='') {
    loadList(unescape(g.filterUrl));
  }
  else {
    loadList();
  }
  showLinks();
  showItems();
  showQueries();
  buildTemplate();
}
```

Collection+JSON

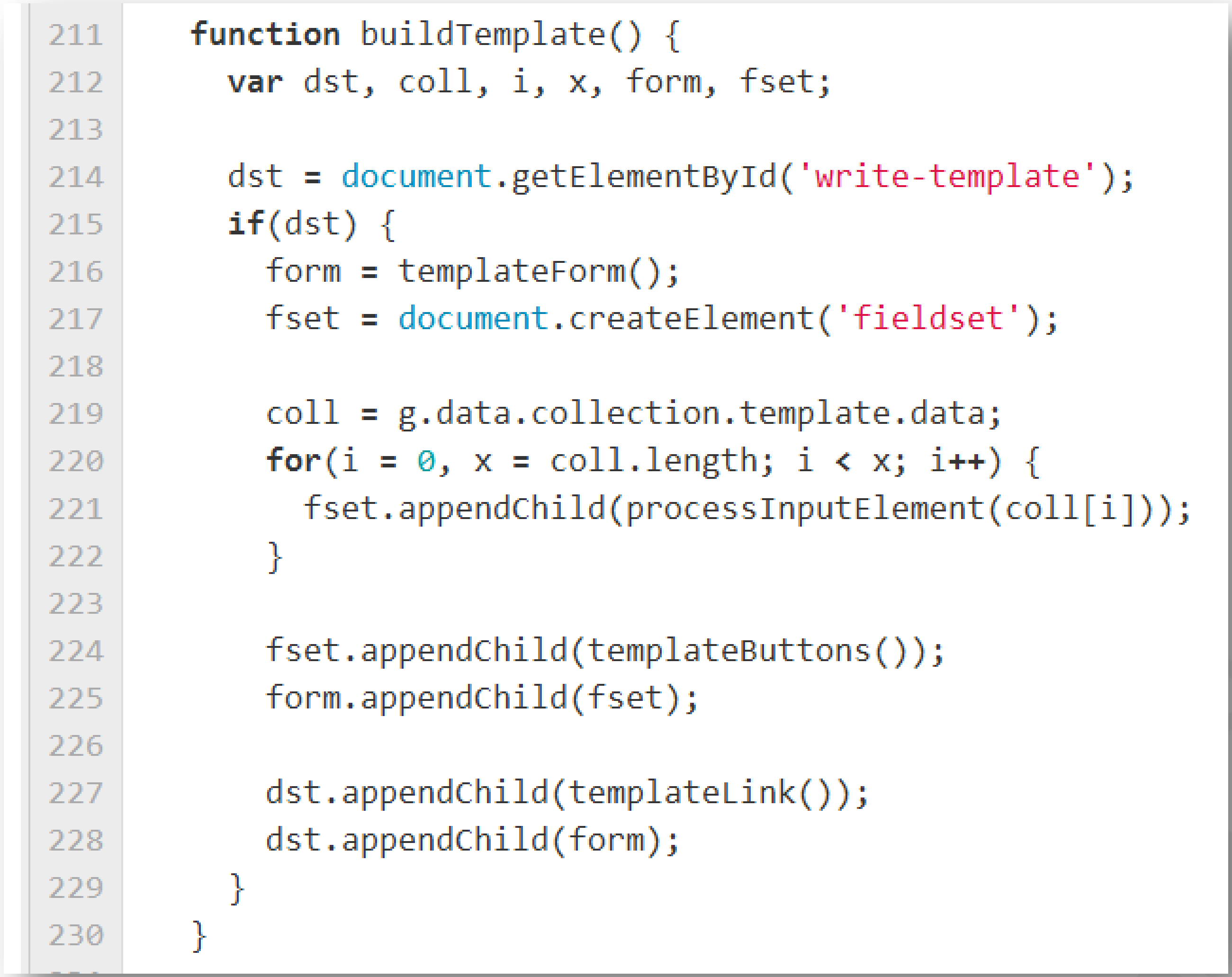
Client Code



```
function processData(coll) {
  var i, x, ul, li, sp;

  ul = document.createElement('ul');

  if(coll) {
    for(i = 0, x = coll.length; i < x; i++) {
      if(coll[i].name && coll[i].value) {
        li = document.createElement('li');
        sp = document.createElement('span');
        sp.className = coll[i].name;
        sp.title = coll[i].name;
        sp.innerHTML = coll[i].value;
        li.appendChild(sp);
        ul.appendChild(li);
      }
    }
  }
  return ul;
}
```

Collection+JSON

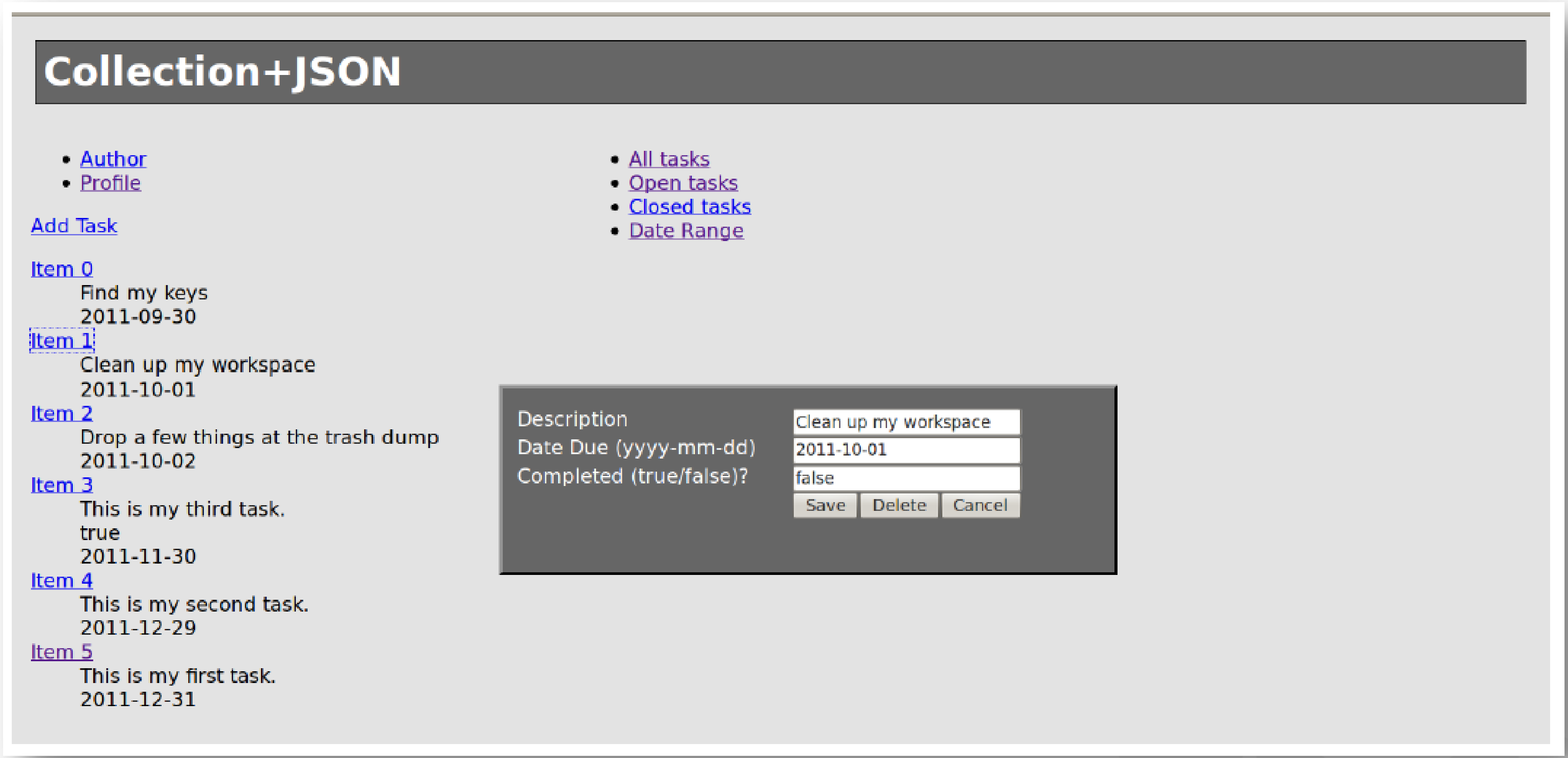
Client Code



```
function buildTemplate() {
  var dst, coll, i, x, form, fset;

  dst = document.getElementById('write-template');
  if(dst) {
    form = templateForm();
    fset = document.createElement('fieldset');

    coll = g.data.collection.template.data;
    for(i = 0, x = coll.length; i < x; i++) {
      fset.appendChild(processInputElement(coll[i]));
    }

    fset.appendChild(templateButtons());
    form.appendChild(fset);

    dst.appendChild(templateLink());
    dst.appendChild(form);
  }
}
```

Collection+JSON

Client App


Collection+JSON

- Author
- Profile

Add Task

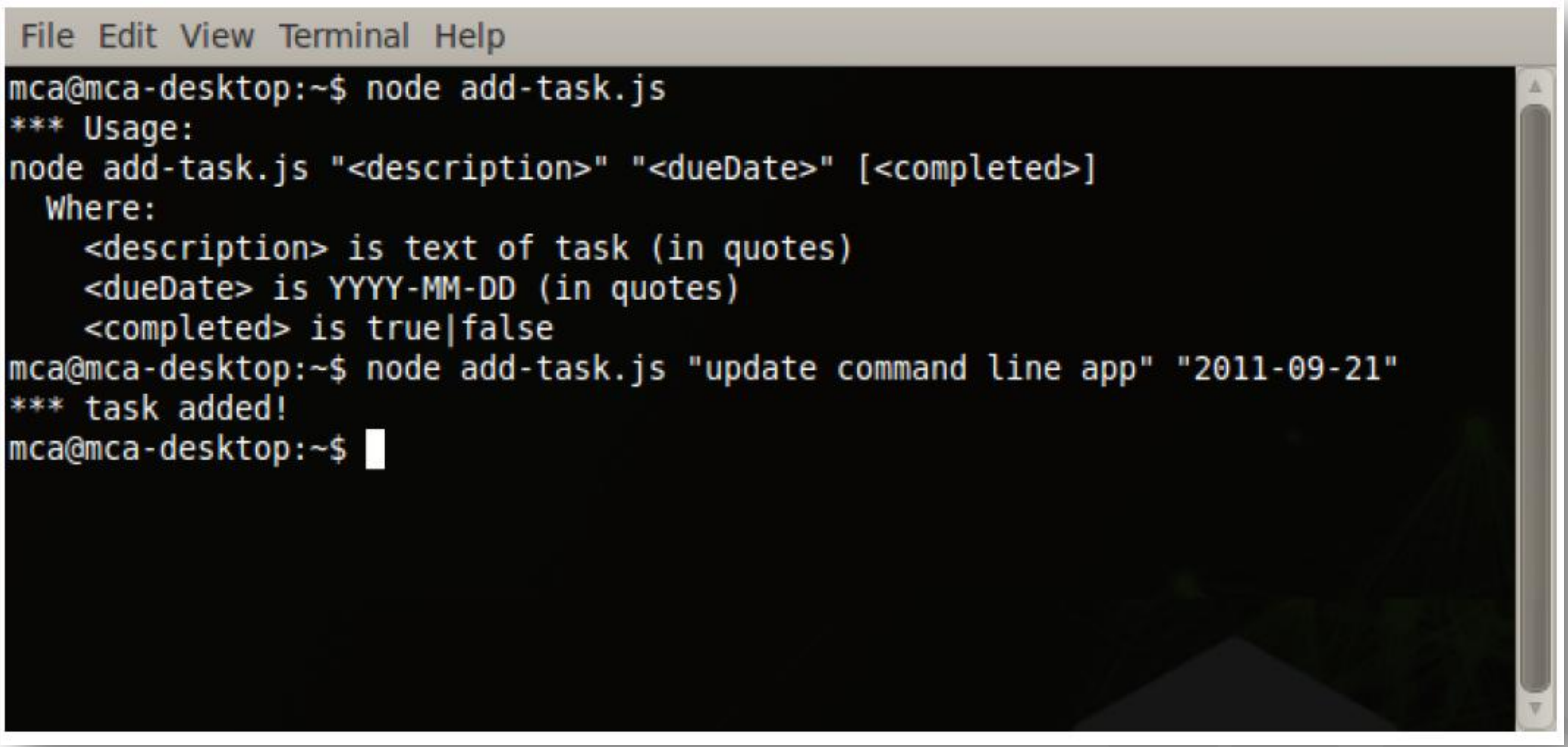
Item 0
    Find my keys
    2011-09-30
Item 1
    Clean up my workspace
    2011-10-01
Item 2
    Drop a few things at the trash dump
    2011-10-02
Item 3
    This is my third task.
    true
    2011-11-30
Item 4
    This is my second task.
    2011-12-29
Item 5
    This is my first task.
    2011-12-31

- All tasks
- Open tasks
- Closed tasks
- Date Range

Description	Clean up my workspace
Date Due (yyyy-mm-dd)	2011-10-01
Completed (true/false)?	false

Collection+JSON

Client App



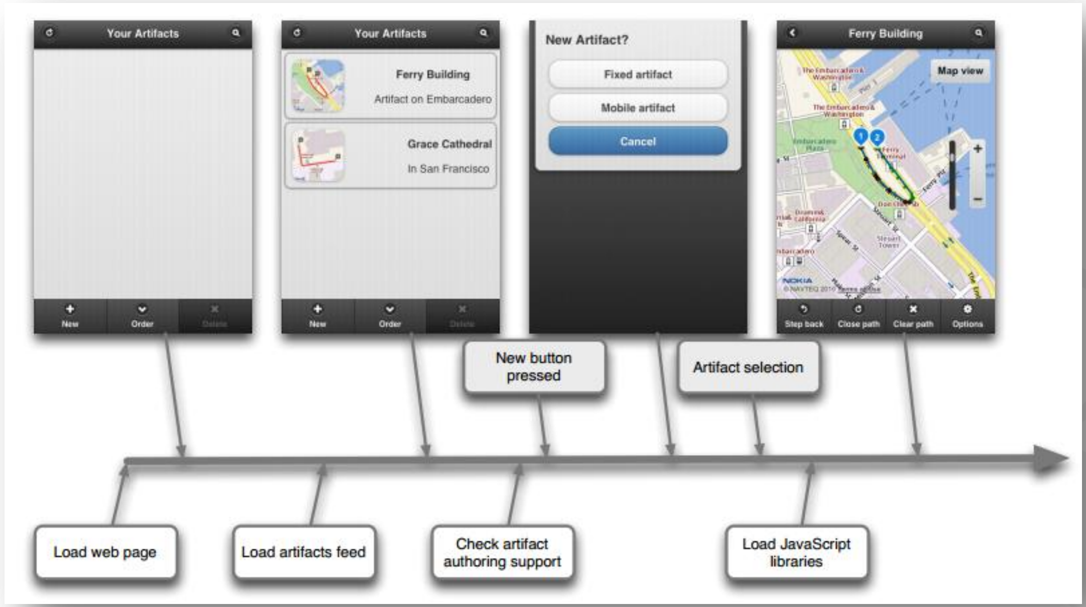
```
mca@mca-desktop:~$ node add-task.js
*** Usage:
node add-task.js "<description>" "<dueDate>" [<completed>]
  Where:
    <description> is text of task (in quotes)
    <dueDate> is YYYY-MM-DD (in quotes)
    <completed> is true|false
mca@mca-desktop:~$ node add-task.js "update command line app" "2011-09-21"
*** task added!
mca@mca-desktop:~$ 
```

Collection+JSON

In the wild…

Collection+JSON

- **Nokia Research - Live Mixed Reality - Vlad Stribu**
- *“[Collection+JSON] … allows us to develop authoring tools that have the ability to self-adapt the user interface to the usage context.”*

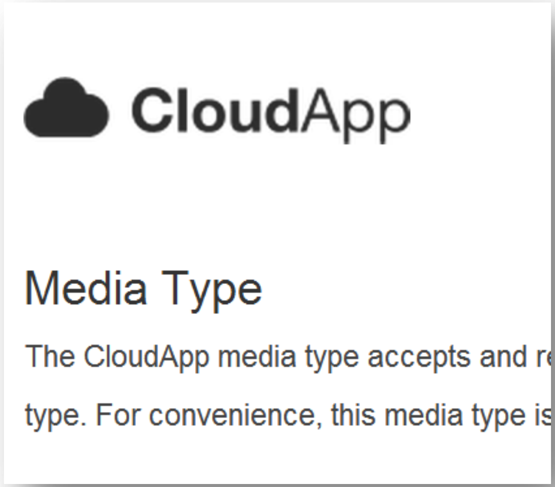


Collection+JSON

- **Nokia Research - Live Mixed Reality - Vlad Stribu**
- *"Collection+JSON was close enough to what we were looking for…"*
- *"Most important factor that influenced our decision was the community around this format…"*

Collection+JSON

- **CloudApp – Larry Marburger**
- *“CloudApp allows you to share images, links, music, videos and files.”*
- *“[Due to developer team changes] we had some setbacks with the Mac app and subsequently the API. We just started working with another developer who's making amazing progress.”*


CloudApp

Media Type

The CloudApp media type accepts and re

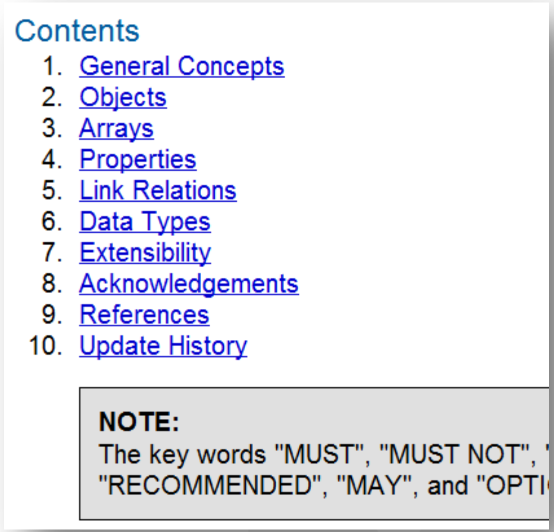
type. For convenience, this media type is

Collection+JSON

- **ember.js – Yehuda Katz**
- *“A framework for creating ambitious applications.”*
- *“By default, it's somewhat repetitive, but that can be addressed…”*
- *“It was straight-forward to extend it with features I needed”*
- *“I am starting to feel like with the number of extensions, I should consider [creating] my own media type.”*

Collection+JSON

- Characteristics
  - Read/Write w/ Templates
  - Small set of link identifiers
  - General “List Domain” handler
- Benefits
  - Limited design means simple parser
  - Servers easy, clients harder
  - Built-in support for custom domain annotations
- Costs
  - Domain mapping is more difficult
  - M2M clients limited to pre-declared vocabulary


Contents

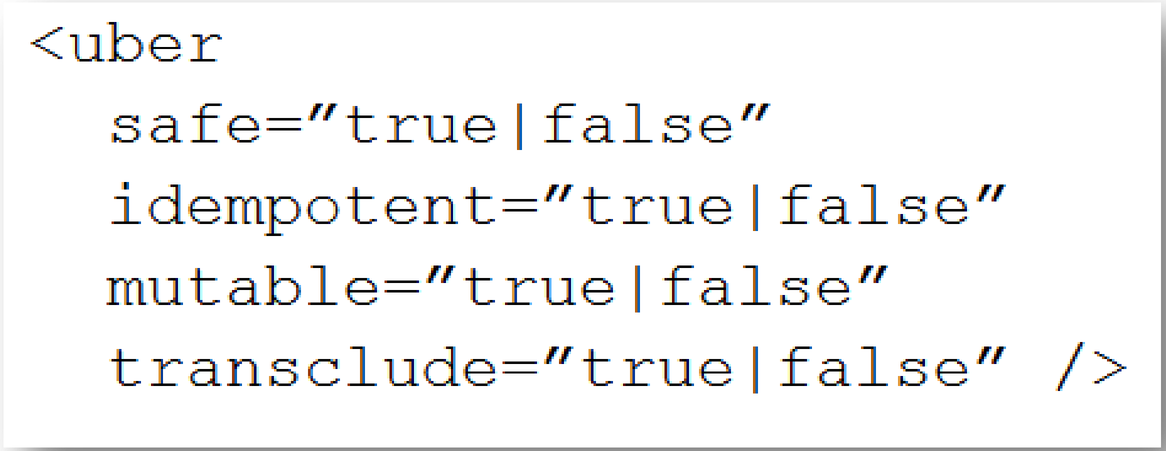
NOTE:
The key words "MUST", "MUST NOT", "
"RECOMMENDED", "MAY", and "OPTI

Affordance Aspects

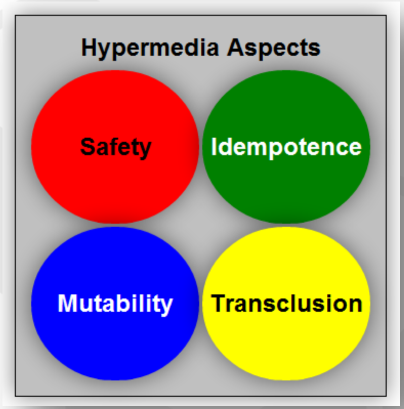
“So, are all affordances essentially the same?”

Affordance Aspects

- *“For the purposes of applying affordances to hypermedia, there are four important aspects to consider”*

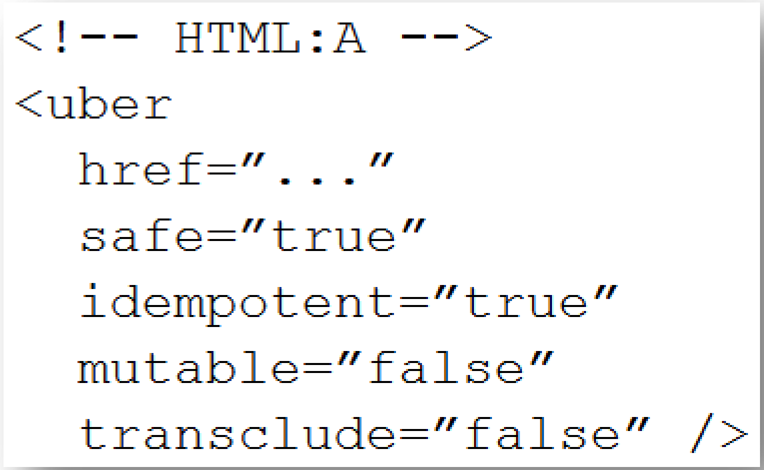


```
<uber
   safe="true|false"
   idempotent="true|false"
   mutable="true|false"
   transclude="true|false" />
```

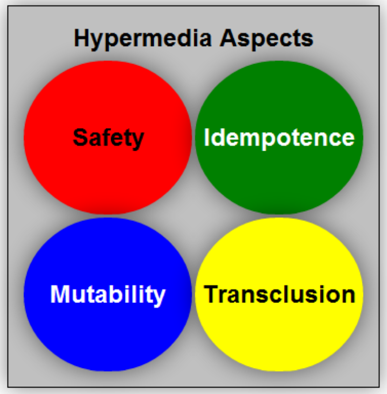


Affordance Aspects

- **Safe**
- The HTTP protocol supports a number of "safe" actions such as HEAD, and GET.



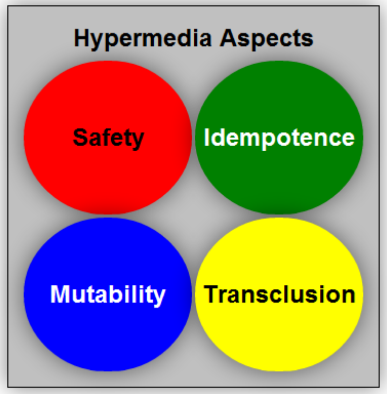
```
<!-- HTML:A -->
<uber
  href="..."
  safe="true"
  idempotent="true"
  mutable="false"
  transclude="false" />
```



Affordance Aspects

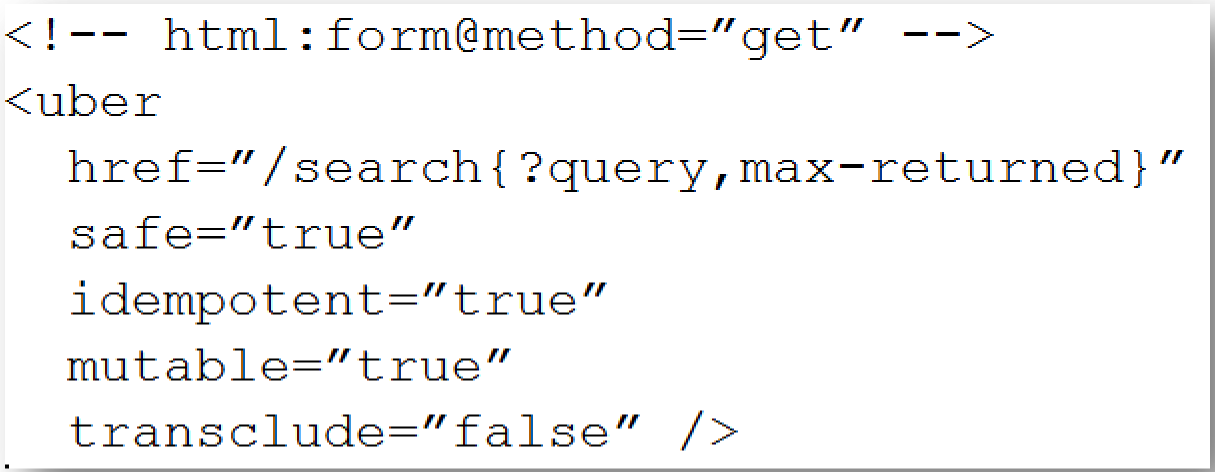
- **Safe**
- The HTTP protocol supports a number of "safe" actions such as HEAD, and GET.
- The HTTP methods PUT, POST, and DELETE are categorized as "unsafe" actions.

```
<!-- atom:link@rel="edit" -->
<uber
  href="..."
  safe="false"
  idempotent="true"
  mutable="false"
  transclude="false" />
```

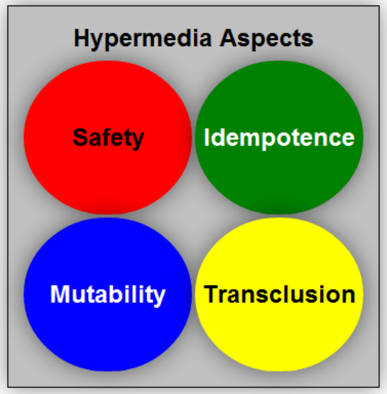


Affordance Aspects

- **Idempotent**
- When an HTML:FORM element has the METHOD property set to "get" it represents an idempotent action.

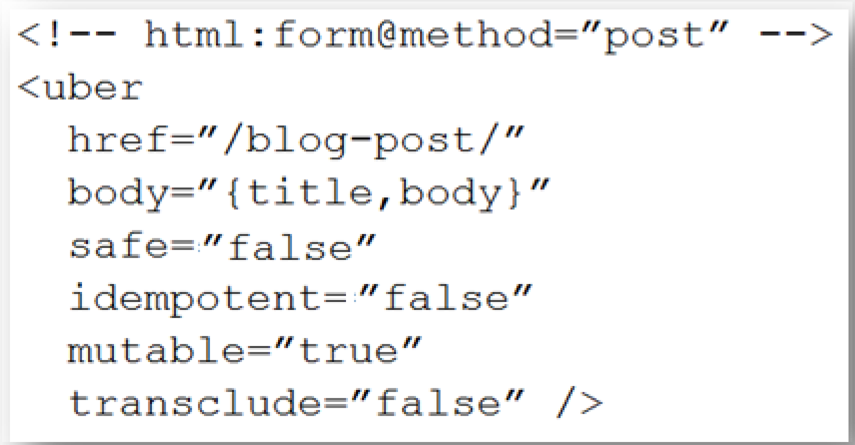


```
<!-- html:form@method="get" -->
<uber
   href="/search{?query,max-returned}"
   safe="true"
   idempotent="true"
   mutable="true"
   transclude="false" />
```

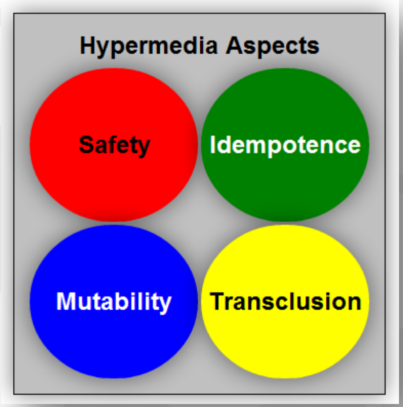


Affordance Aspects

- **Idempotent**
- When an HTML:FORM element has the METHOD property set to "get" it represents an idempotent action.
- When the same property is set to "post" the affordance represents a non-idempotent action.

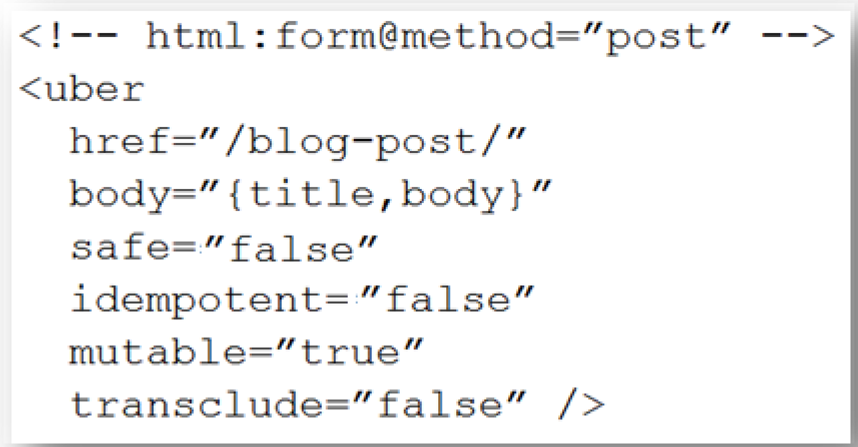


```
<!-- html:form@method="post" -->
<uber
  href="/blog-post/"
  body="{title,body}"
  safe="false"
  idempotent="false"
  mutable="true"
  transclude="false" />
```

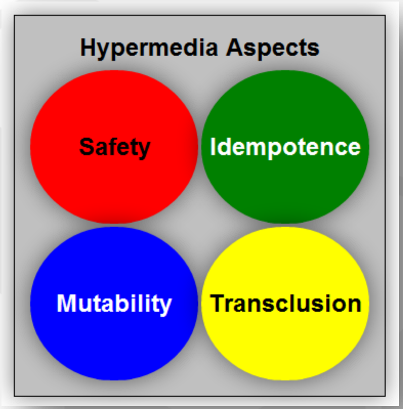


Affordance Aspects

- **Mutability**
- HTML:FORM affords mutability

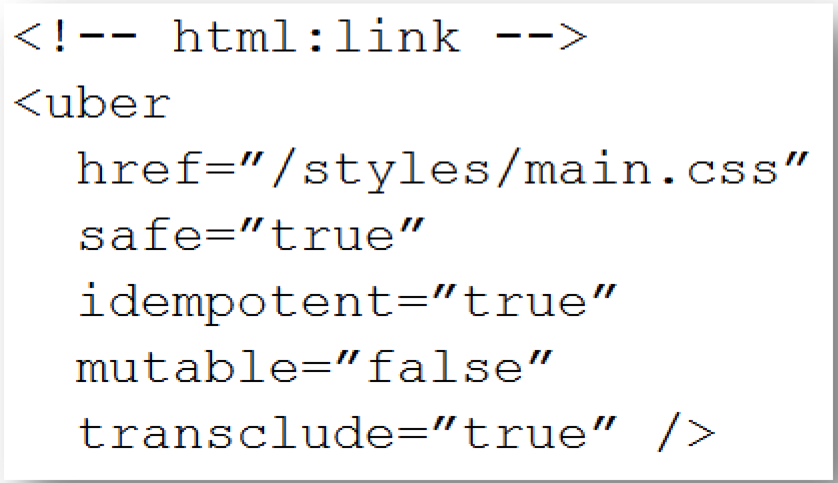


```
<!-- html:form@method="post" -->
<uber
  href="/blog-post/"
  body="{title,body}"
  safe="false"
  idempotent="false"
  mutable="true"
  transclude="false" />
```

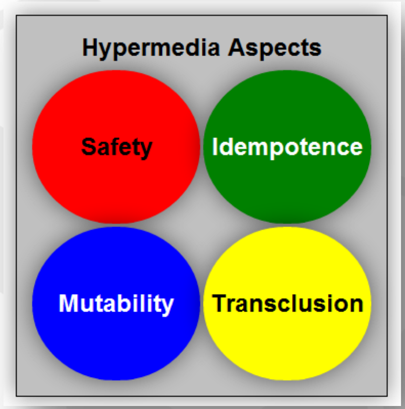


Affordance Aspects

- **Mutability**
- HTML:FORM affords mutability
- HTML:LINK is immutable



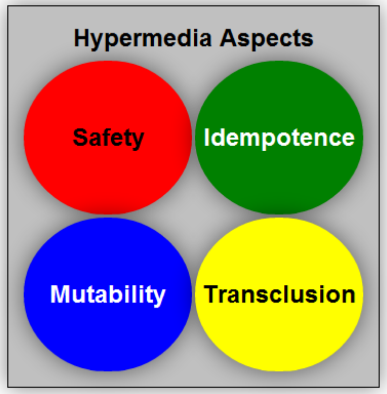
```
<!-- html:link -->
<uber
  href="/styles/main.css"
  safe="true"
  idempotent="true"
  mutable="false"
  transclude="true" />
```



Affordance Aspects

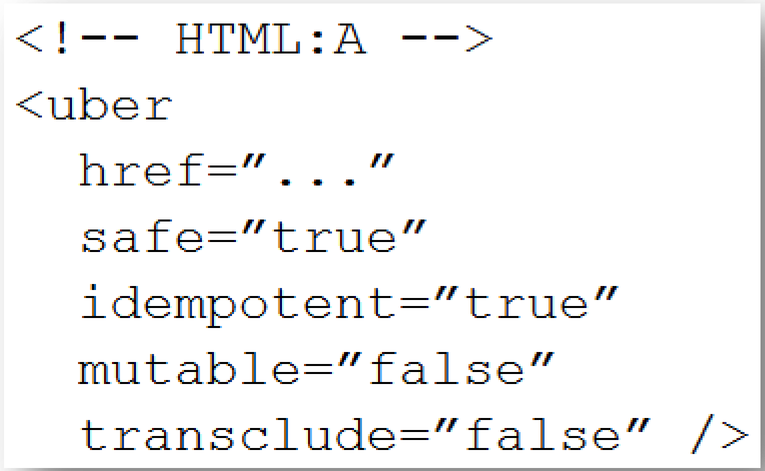
- **Transclusion**
- HTML:IMG affords transclusion

```
<!-- html:img -->
<uber
  href="/images/logo.png"
  safe="true"
  idempotent="true"
  mutable="false"
  transclude="true" />
```

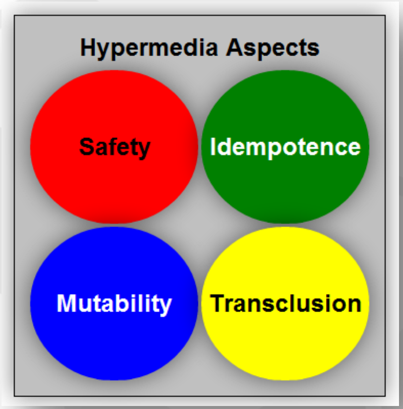


Affordance Aspects

- **Transclusion**
- HTML:IMG affords transclusion
- HTML:A does not

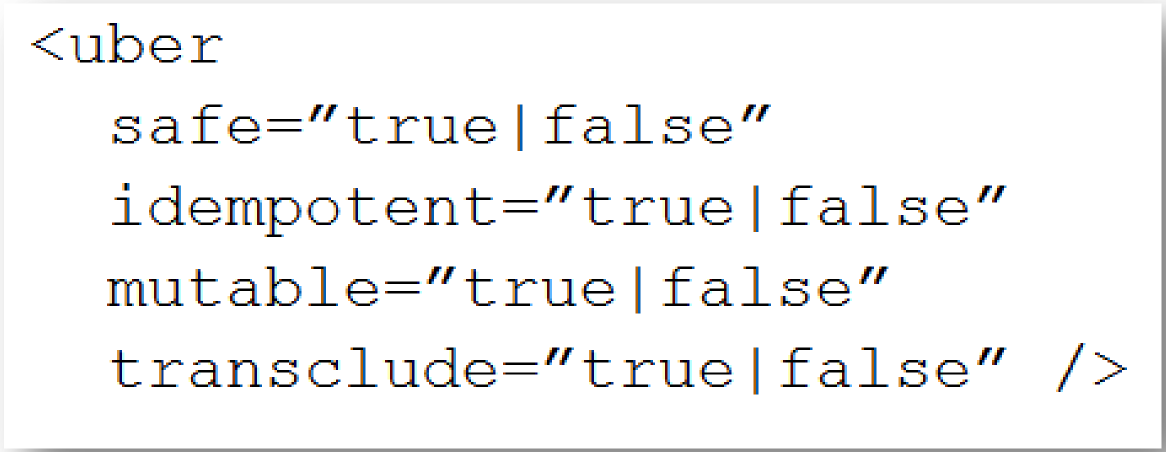


```
<!-- HTML:A -->
<uber
   href="..."
   safe="true"
   idempotent="true"
   mutable="false"
   transclude="false" />
```

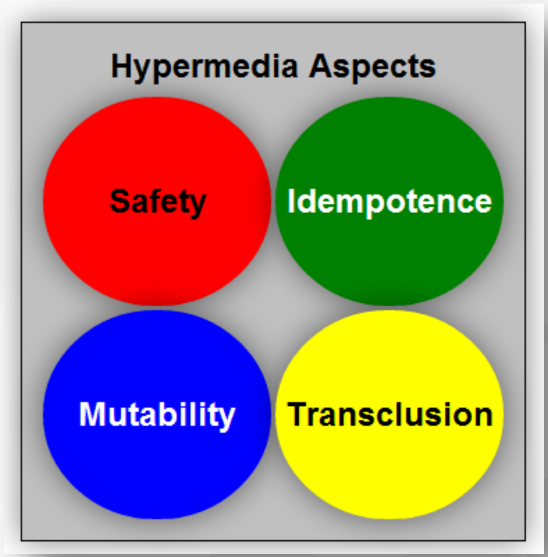


Affordance Aspects

- However, this single affordance is not very usable.



```
<uber
   safe="true|false"
   idempotent="true|false"
   mutable="true|false"
   transclude="true|false" />
```



Affordance Aspects

Media types should be usable

Affordance Aspects

Messages should be usable

Affordance Aspects

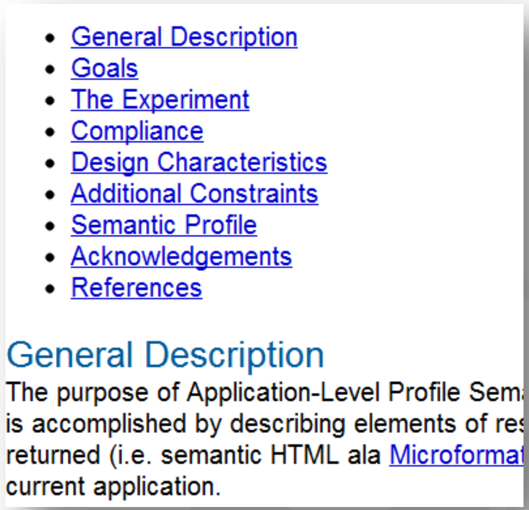
APIs should be usable

ALPS for HTML

Design #3: Aspects, Factors, Abstractions

ALPS for HTML

- **ALPS profile URI**
- First designs in early 2011 (not registered)
- *“The purpose of Application-Level Profile Semantics (ALPS) is to document the application-level semantics of a particular implementation.”*
- *“The example profile here contains details on customizing the XHTML media type for a specific application domain: **Micro-blogging**.”*



- General Description
- Goals
- The Experiment
- Compliance
- Design Characteristics
- Additional Constraints
- Semantic Profile
- Acknowledgements
- References

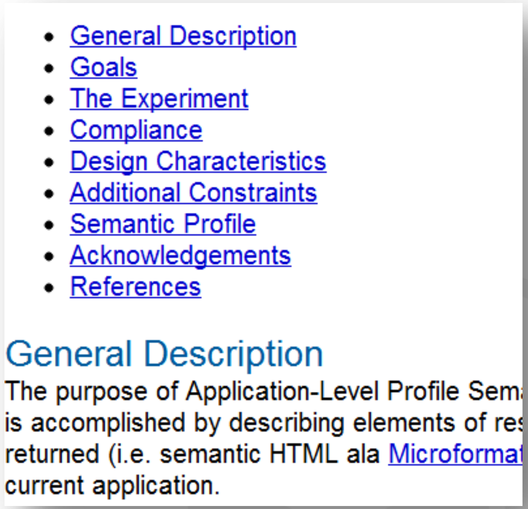
General Description

The purpose of Application-Level Profile Sem
is accomplished by describing elements of res
returned (i.e. semantic HTML ala Microformat
current application.

LAYER 7 TECHNOLOGIES

ALPS for HTML

- **ALPS profile URI**
- Multiple parties building their own client or server applications without seeing each other's work or accessing a running "reference" implementation.
- Developers are expected to rely on the constraints and definitions found in this document (and the referenced RFCs) as the sole instruction.



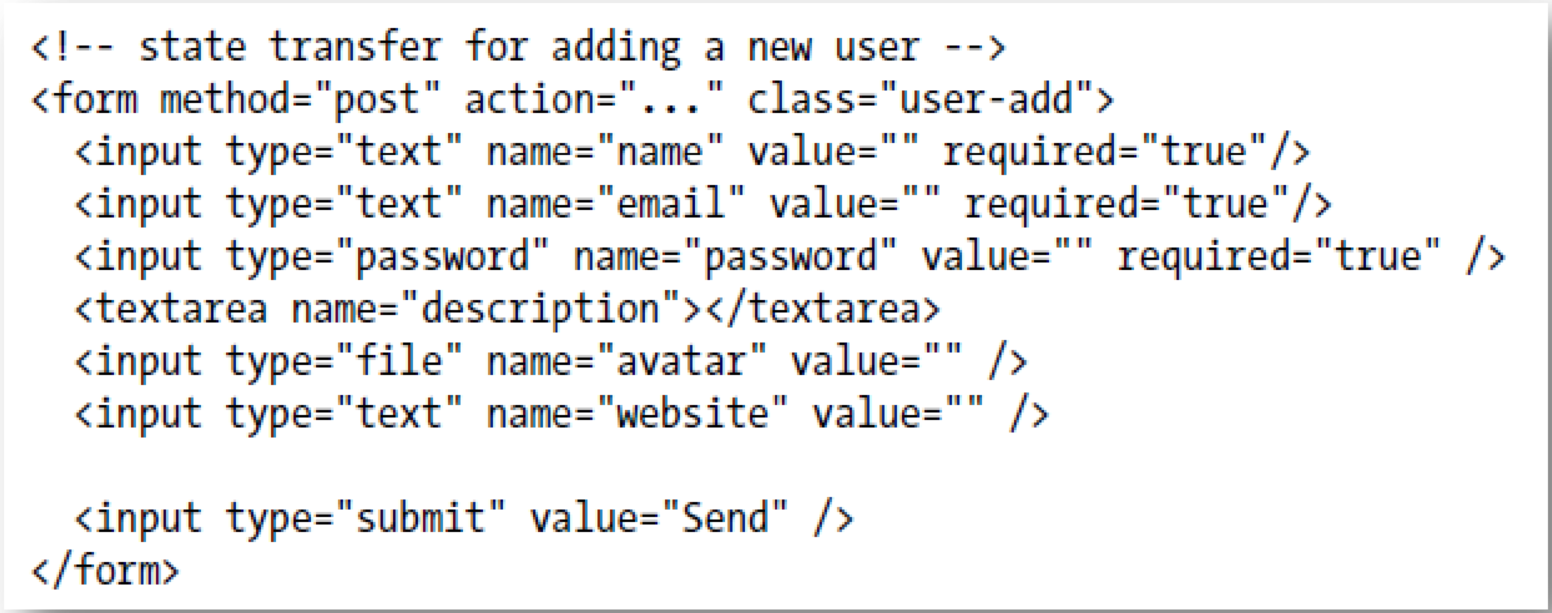
- General Description
- Goals
- The Experiment
- Compliance
- Design Characteristics
- Additional Constraints
- Semantic Profile
- Acknowledgements
- References

General Description

The purpose of Application-Level Profile Sem
is accomplished by describing elements of res
returned (i.e. semantic HTML ala Microformat
current application.

ALPS for HTML

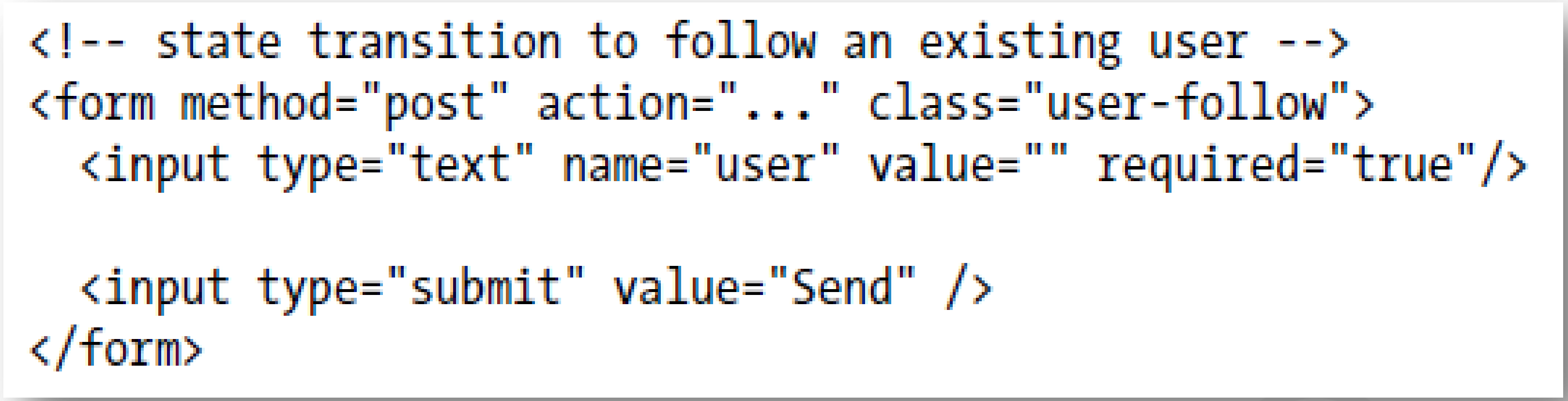
Messages



```
<!-- state transfer for adding a new user -->
<form method="post" action="..." class="user-add">
  <input type="text" name="name" value="" required="true"/>
  <input type="text" name="email" value="" required="true"/>
  <input type="password" name="password" value="" required="true" />
  <textarea name="description"></textarea>
  <input type="file" name="avatar" value="" />
  <input type="text" name="website" value="" />

  <input type="submit" value="Send" />
</form>
```

ALPS for HTML

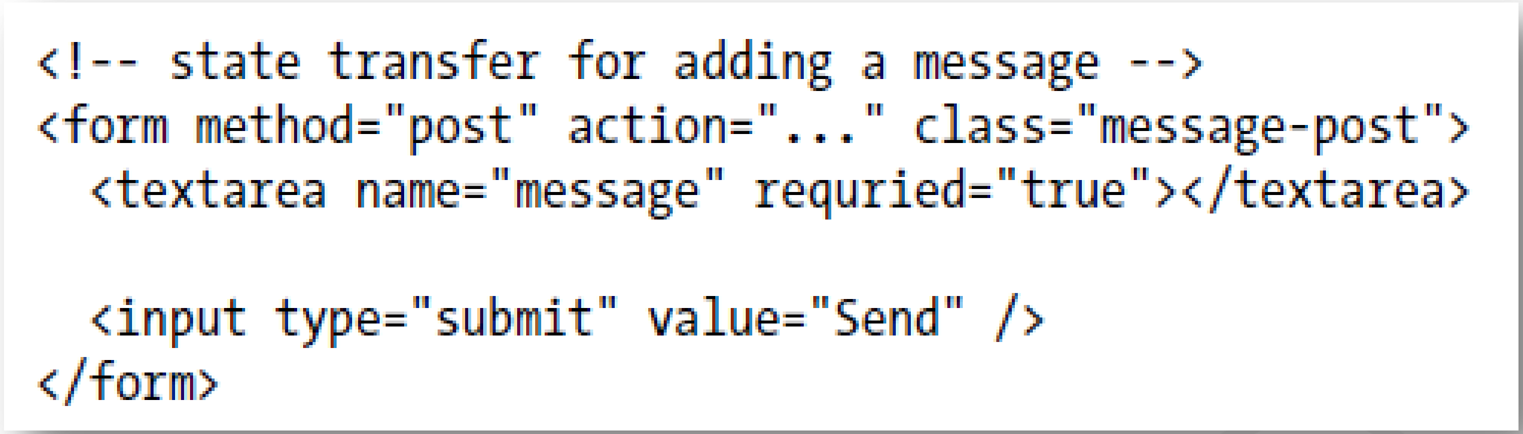
Messages



```
<!-- state transition to follow an existing user -->
<form method="post" action="..." class="user-follow">
  <input type="text" name="user" value="" required="true"/>

  <input type="submit" value="Send" />
</form>
```

ALPS for HTML

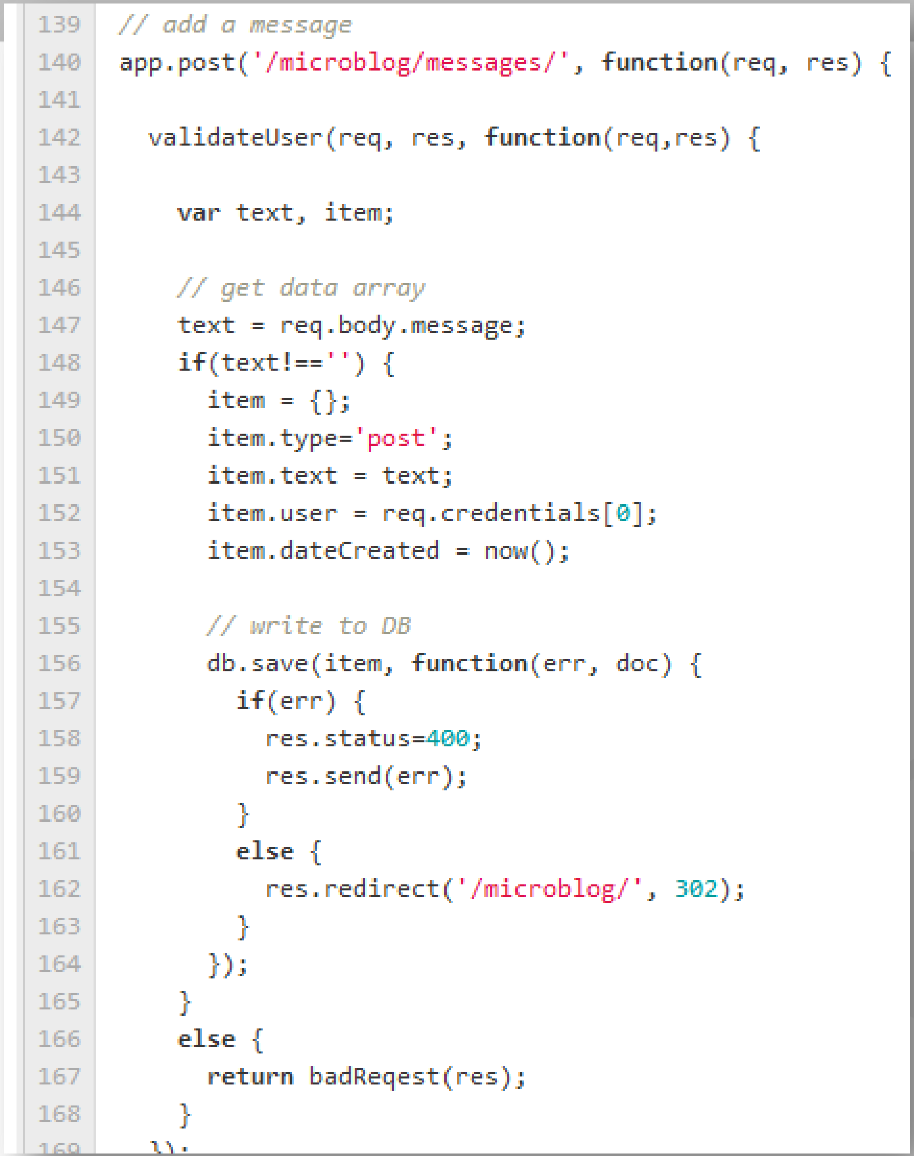
Messages



```
<!-- state transfer for adding a message -->
<form method="post" action="..." class="message-post">
  <textarea name="message" requried="true"></textarea>

  <input type="submit" value="Send" />
</form>
```

ALPS for HTML

Server Code



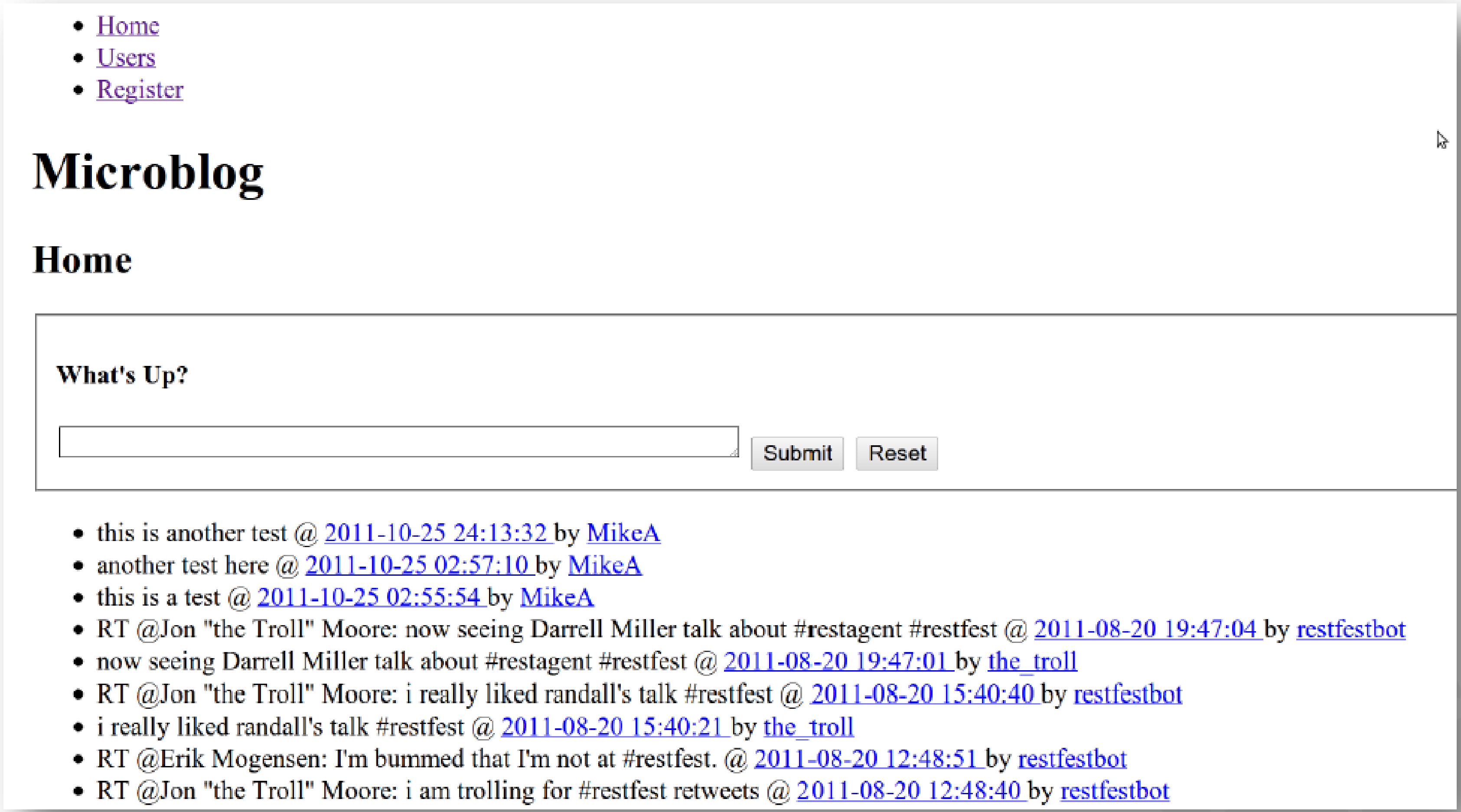
```
// add a message
app.post('/microblog/messages/', function(req, res) {

  validateUser(req, res, function(req,res) {

    var text, item;

    // get data array
    text = req.body.message;
    if(text!=='') {
      item = {};
      item.type='post';
      item.text = text;
      item.user = req.credentials[0];
      item.dateCreated = now();

      // write to DB
      db.save(item, function(err, doc) {
        if(err) {
          res.status=400;
          res.send(err);
        }
        else {
          res.redirect('/microblog/', 302);
        }
      });
    }
    else {
      return badReqest(res);
    }
  });
```

ALPS for HTML

Client Code

ALPS for HTML

Client App



- Home
- Users
- Register

Microblog

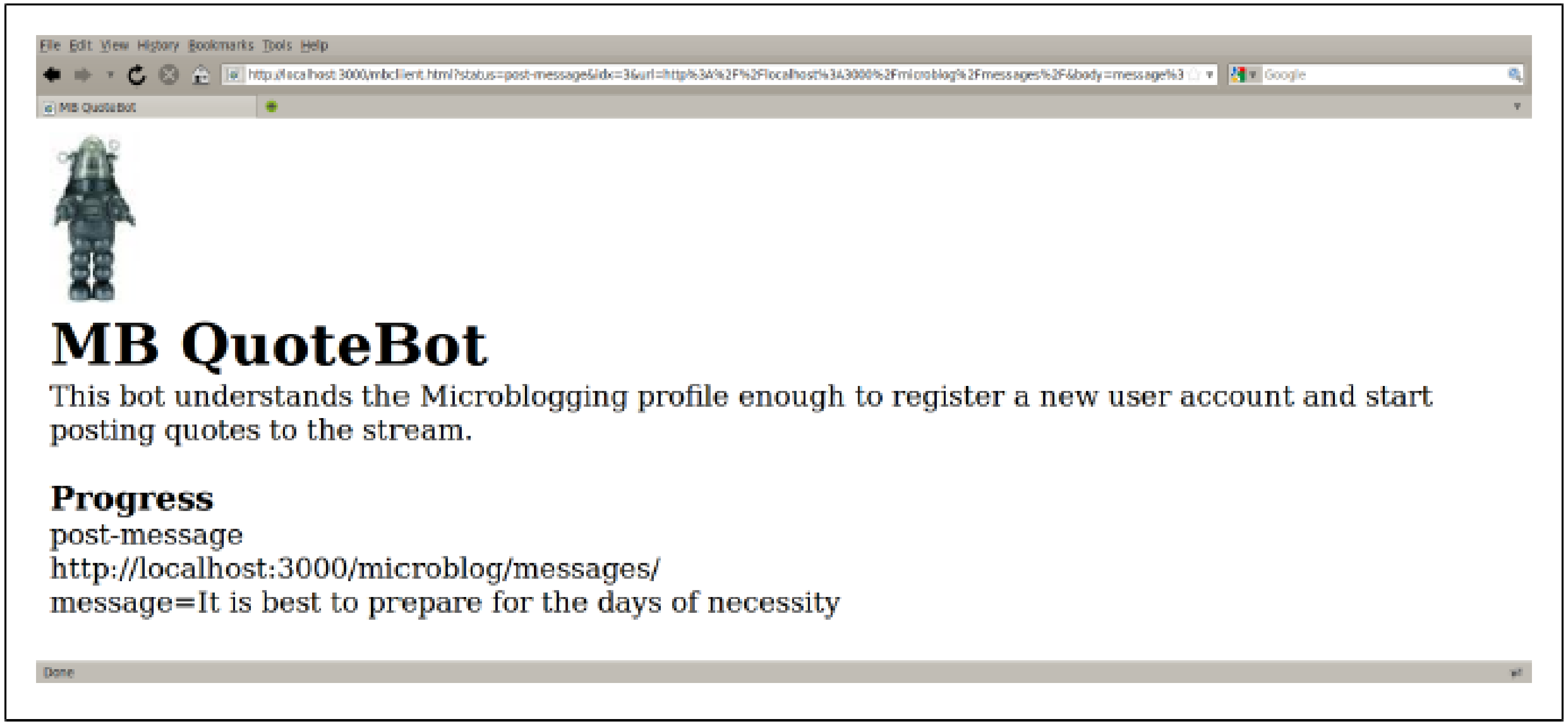
Home

What's Up?

- this is another test @ 2011-10-25 24:13:32 by MikeA
- another test here @ 2011-10-25 02:57:10 by MikeA
- this is a test @ 2011-10-25 02:55:54 by MikeA
- RT @Jon "the Troll" Moore: now seeing Darrell Miller talk about #restagent #restfest @ 2011-08-20 19:47:04 by restfestbot
- now seeing Darrell Miller talk about #restagent #restfest @ 2011-08-20 19:47:01 by the_troll
- RT @Jon "the Troll" Moore: i really liked randall's talk #restfest @ 2011-08-20 15:40:40 by restfestbot
- i really liked randall's talk #restfest @ 2011-08-20 15:40:21 by the_troll
- RT @Erik Mogensen: I'm bummed that I'm not at #restfest. @ 2011-08-20 12:48:51 by restfestbot
- RT @Jon "the Troll" Moore: i am trolling for #restfest retweets @ 2011-08-20 12:48:40 by restfestbot

ALPS for HTML

Client Bot


File Edit View History Bookmarks Tools Help
http://localhost:3000/mbclient.html?status=post-message&idx=3&url=http%3A%2F%2Flocalhost%3A3000%2Fmicroblog%2Fmessages%2F&body=message%3
Google
MB QuoteBot
Done

MB QuoteBot

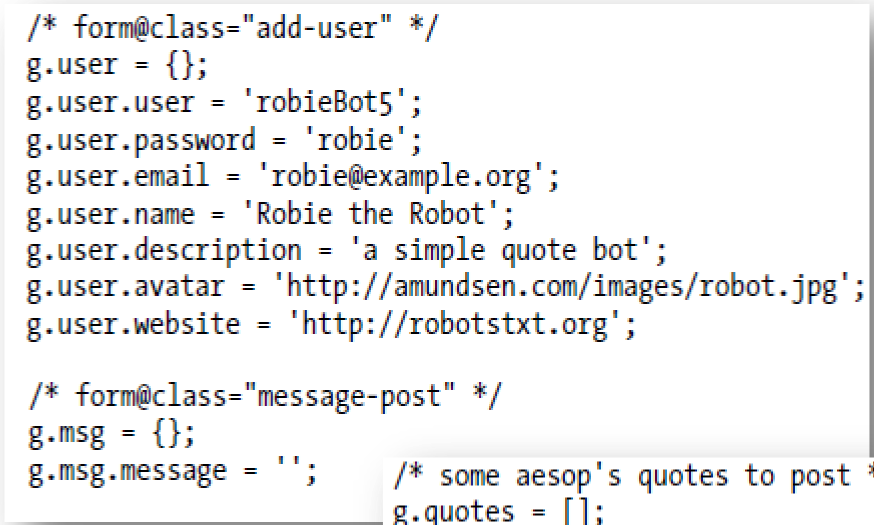
This bot understands the Microblogging profile enough to register a new user account and start posting quotes to the stream.

Progress

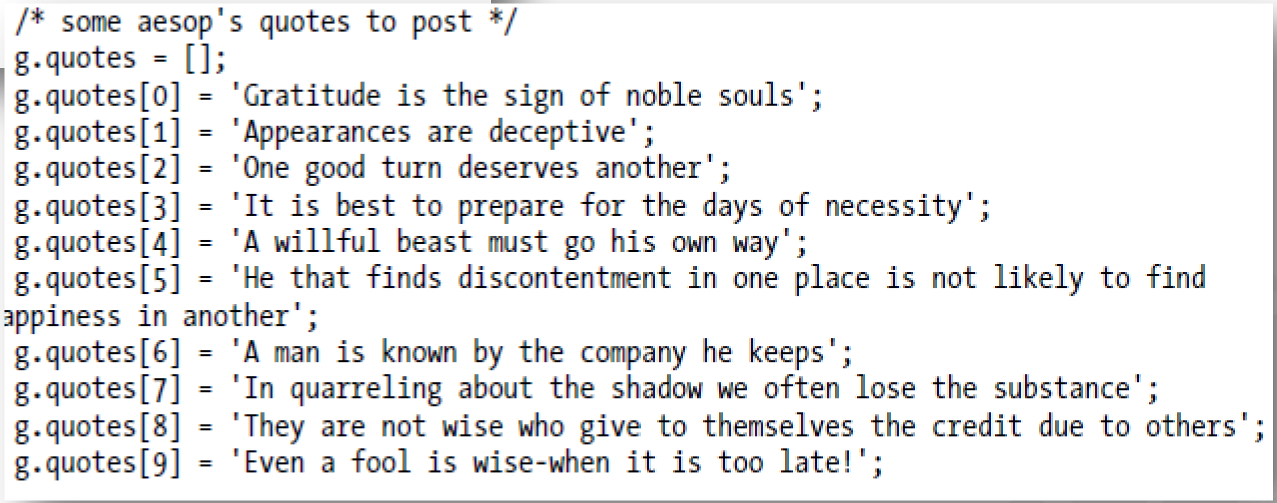
post-message
http://localhost:3000/microblog/messages/
message=It is best to prepare for the days of necessity

ALPS for HTML

Client Bot



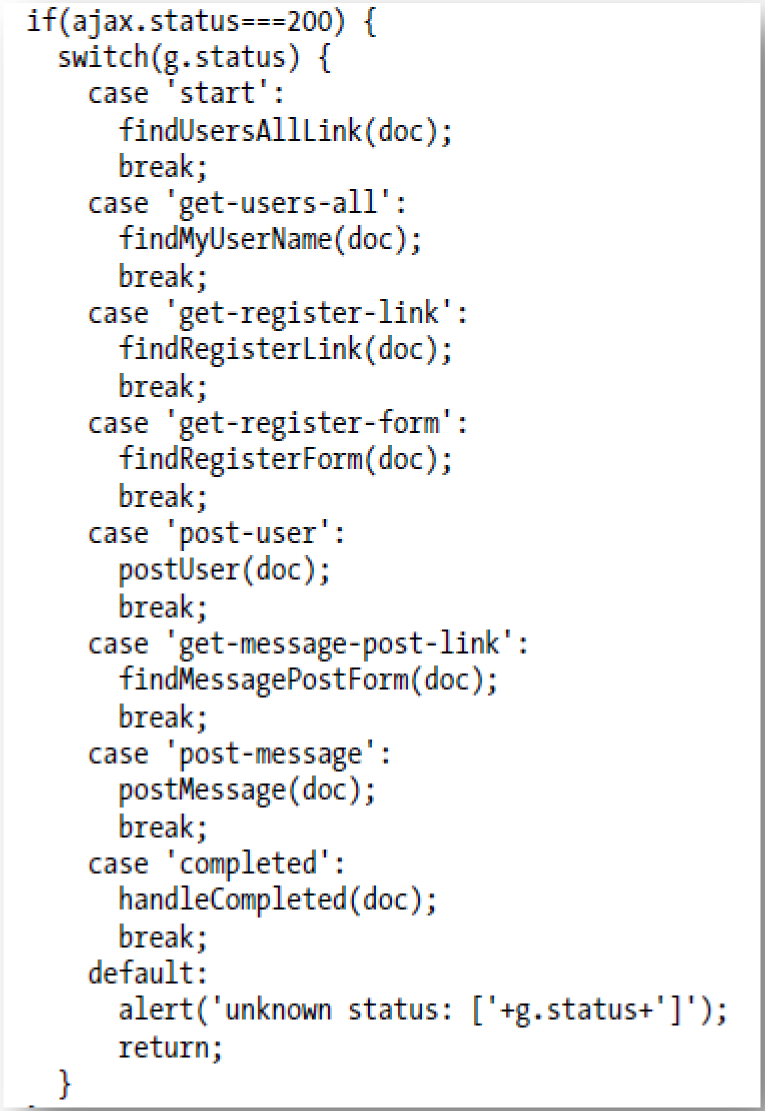
```
/* form@class="add-user" */
g.user = {};
g.user.user = 'robieBot5';
g.user.password = 'robie';
g.user.email = 'robie@example.org';
g.user.name = 'Robie the Robot';
g.user.description = 'a simple quote bot';
g.user.avatar = 'http://amundsen.com/images/robot.jpg';
g.user.website = 'http://robotstxt.org';

/* form@class="message-post" */
g.msg = {};
g.msg.message = '';
```



```
/* some aesop's quotes to post */
g.quotes = [];
g.quotes[0] = 'Gratitude is the sign of noble souls';
g.quotes[1] = 'Appearances are deceptive';
g.quotes[2] = 'One good turn deserves another';
g.quotes[3] = 'It is best to prepare for the days of necessity';
g.quotes[4] = 'A willful beast must go his own way';
g.quotes[5] = 'He that finds discontentment in one place is not likely to find
appiness in another';
g.quotes[6] = 'A man is known by the company he keeps';
g.quotes[7] = 'In quarreling about the shadow we often lose the substance';
g.quotes[8] = 'They are not wise who give to themselves the credit due to others';
g.quotes[9] = 'Even a fool is wise-when it is too late!';
```

ALPS for HTML

Client Bot



```
if(ajax.status===200) {
  switch(g.status) {
    case 'start':
      findUsersAllLink(doc);
      break;
    case 'get-users-all':
      findMyUserName(doc);
      break;
    case 'get-register-link':
      findRegisterLink(doc);
      break;
    case 'get-register-form':
      findRegisterForm(doc);
      break;
    case 'post-user':
      postUser(doc);
      break;
    case 'get-message-post-link':
      findMessagePostForm(doc);
      break;
    case 'post-message':
      postMessage(doc);
      break;
    case 'completed':
      handleCompleted(doc);
      break;
    default:
      alert('unknown status: ['+g.status+']');
      return;
  }
```

ALPS for HTML

“In the wild…”

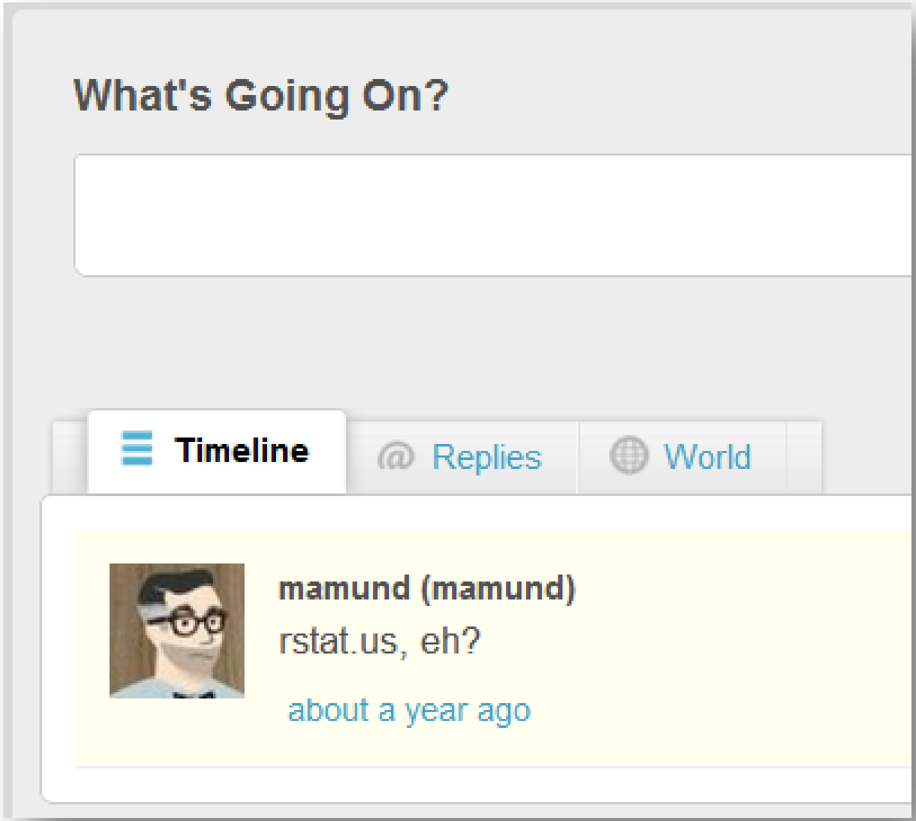
ALPS for HTML

Rstat.us – Carol Nichols

- *“There are two things that make rstat.us special:* ***simplicity*** *and* ***openness****.”*
- *“[S]ince we already have a full-functioning end-user facing site, the ALPS microblogging spec means adding a few attributes rather than having to maintain a totally separate API interface.”*
- “*The current way of presenting the ALPS spec is [too] flat.”*

ALPS for HTML

Rstat.us – Carol Nichols



What's Going On?

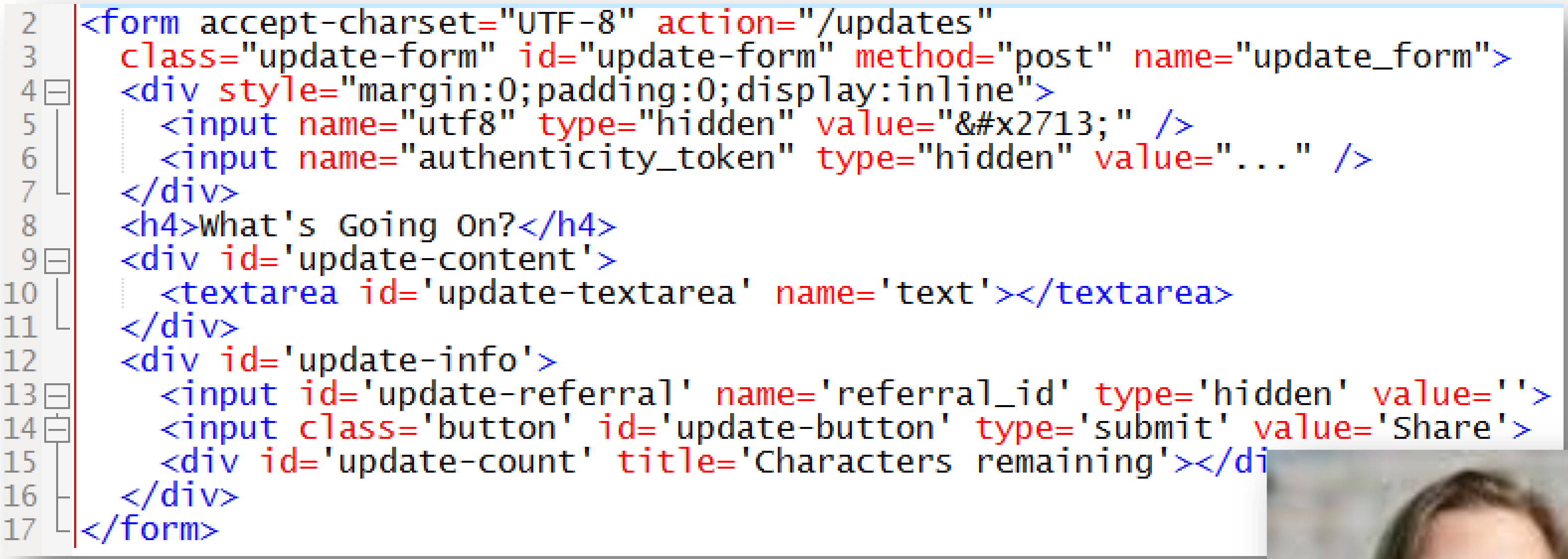
mamund (mamund)

rstat.us, eh?

about a year ago

ALPS for HTML

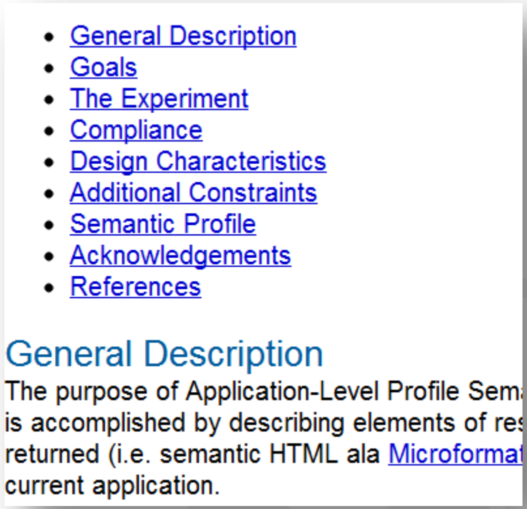
Rstat.us – Carol Nichols



```
<form accept-charset="UTF-8" action="/updates"
  class="update-form" id="update-form" method="post" name="update_form">
  <div style="margin:0;padding:0;display:inline">
    <input name="utf8" type="hidden" value="&#x2713;" />
    <input name="authenticity_token" type="hidden" value="..." />
  </div>
  <h4>What's Going On?</h4>
  <div id='update-content'>
    <textarea id='update-textarea' name='text'></textarea>
  </div>
  <div id='update-info'>
    <input id='update-referral' name='referral_id' type='hidden' value=''>
    <input class='button' id='update-button' type='submit' value='Share'>
    <div id='update-count' title='Characters remaining'></di
  </div>
</form>
```

ALPS for HTML

- Characteristics
  - Domain Semantics Only
  - Media-type agnostic
- Benefits
  - Focused on problem domain
  - Treats “pages” as the “API”
- Costs
  - Very abstract model
  - Tough to document
  - Seems “over complex” esp. for M2M cases


- General Description
- Goals
- The Experiment
- Compliance
- Design Characteristics
- Additional Constraints
- Semantic Profile
- Acknowledgements
- References

General Description

The purpose of Application-Level Profile Sem
is accomplished by describing elements of res
returned (i.e. semantic HTML ala Microformat
current application.

ALPS for HTML

“If HTML is not the only media-type we need…”

ALPS for HTML

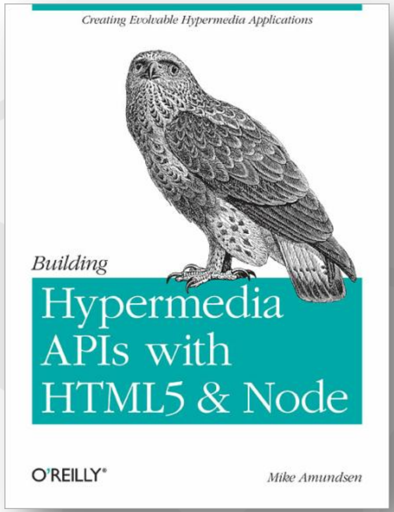
"How do you choose?"

Methodology

Mapping your domain to HTTP

Methodology

- Designing messages is the primary work
- Focus on mapping to payloads, not identifiers
- Survey existing media types first
- If you can't find a suitable H-Factor signature match, consider designing your own.
- *Pro Tip: you can always find a match.*


Creating Evolvable Hypermedia Applications

Building Hypermedia APIs with HTML5 & Node

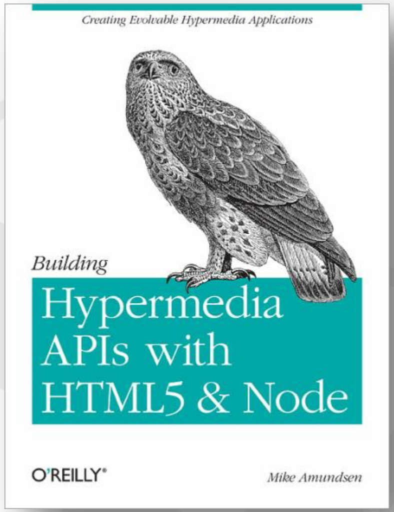
O'REILLY

Mike Amundsen

LAYER 7 TECHNOLOGIES

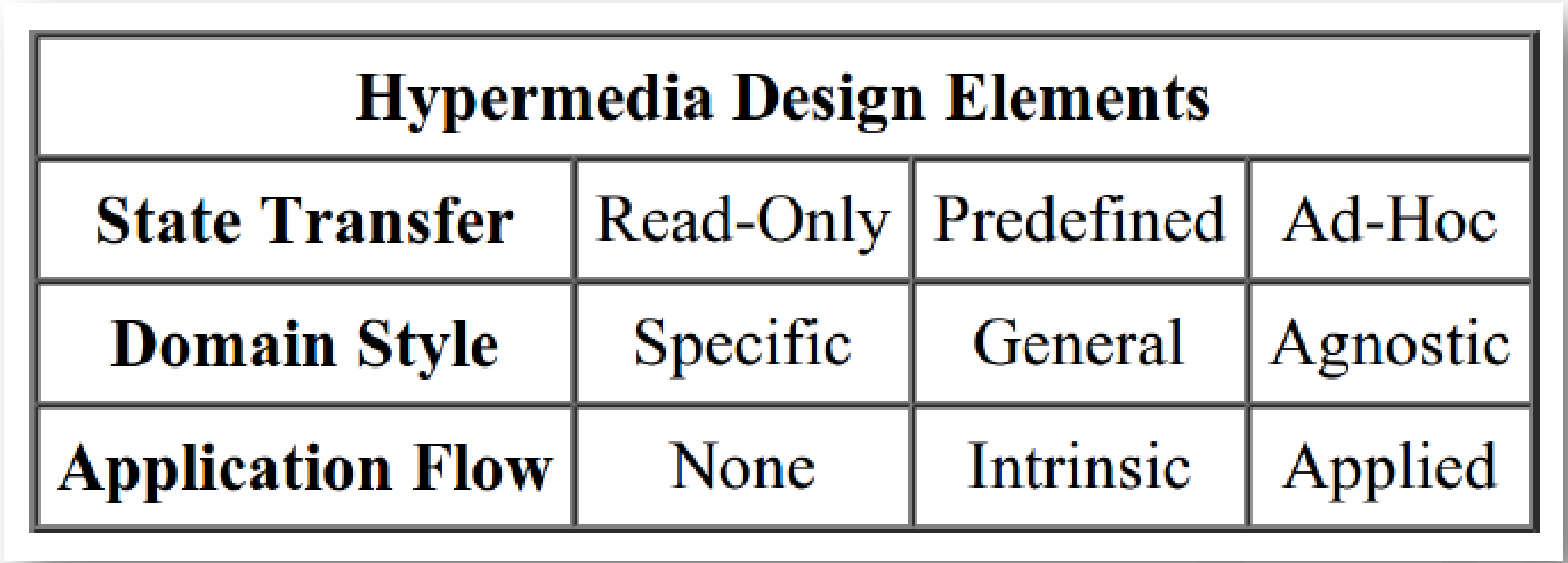
Methodology

- Start with a format (XML, JSON, HTML, etc.)
- You might need to support more than one
- Don’t assume you can “cross-map” formats easily
- *Pro Tip: you almost always need to support more than one.*


Creating Evolvable Hypermedia Applications
Building
Hypermedia APIs with HTML5 & Node
O'REILLY
Mike Amundsen

Methodology

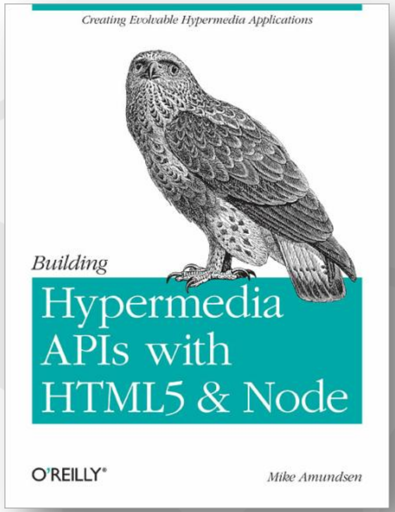
- Select your other design elements as needed



Hypermedia Design Elements			
State Transfer	Read-Only	Predefined	Ad-Hoc
Domain Style	Specific	General	Agnostic
Application Flow	None	Intrinsic	Applied

Methodology

- Represent State, not Objects
- Remember both data and transitions
- Craft lots of messages
- *Pro Tip: you can never have enough messages*

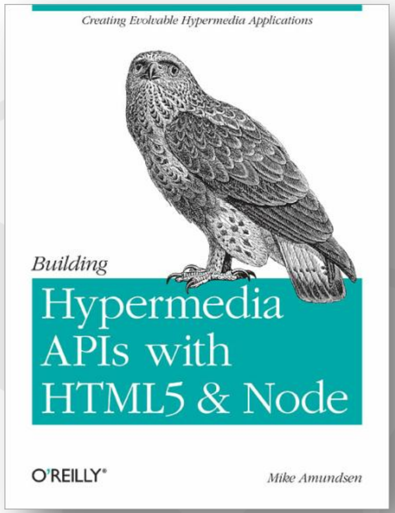

Creating Evolvable Hypermedia Applications
Building
Hypermedia APIs with HTML5 & Node
O'REILLY
Mike Amundsen

Methodology

When you are sure you have:

- The proper format
- The right H-Factor signature
- The correct mapping of domain to messages
- Sufficent message examples

Then, and only then…


Creating Evolvable Hypermedia Applications
Building
Hypermedia APIs with HTML5 & Node
O'REILLY
Mike Amundsen

Methodology

You can start writing the code

Methodology

Because…

Methodology

The code is only
the implementation

Methodology

The code is only
the implementation

The media type is
the design.

Methodology

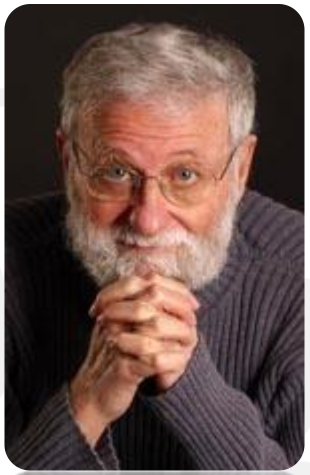
And if you get the design right...

Methodology

Your users will be able to do things

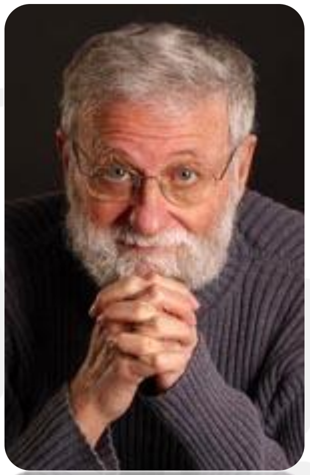
Methodology

Your users will be able to do things



Methodology

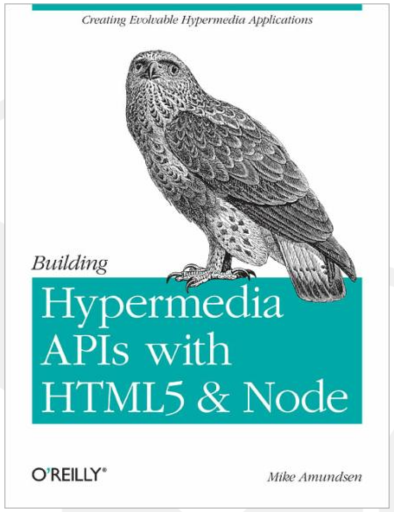
Your users will be able to do things *you never imagined*



Mike Amundsen

The Costs and Benefits of Building Hypermedia APIs (with Node.js)

@mamund


Creating Evolvable Hypermedia Applications
Building
Hypermedia APIs with HTML5 & Node
O'REILLY
Mike Amundsen