# Lively4: Thin Morphic

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

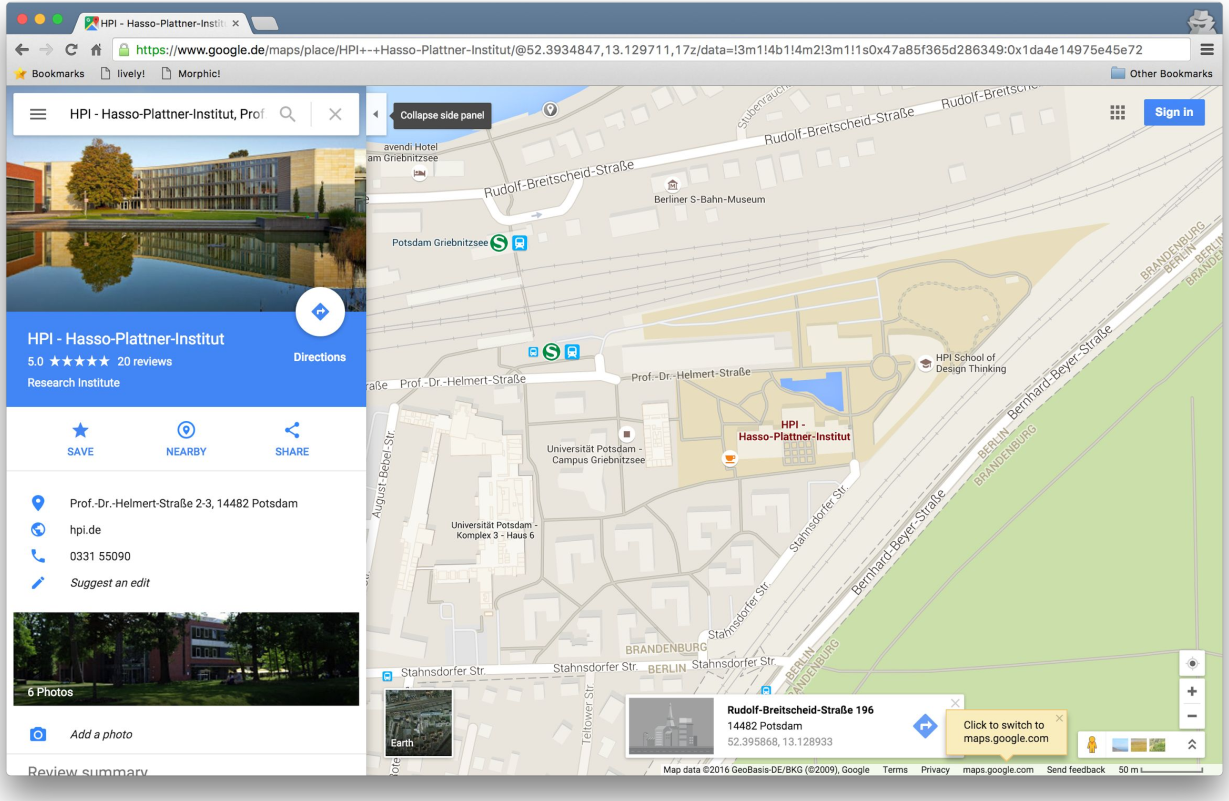
Software Design Seminar 2015/2016
Software Architecture Group

Felix Wolff
Daniel Werner
Astrid Thomschke

# Motivation

# Motivation: What we get

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam



HPI - Hasso-Plattner-Instit ×

https://www.google.de/maps/place/HPI+-+Hasso-Plattner-Institut/@52.3934847,13.129711,17z/data=!3m1!4b1!4m2!3m1!1s0x47a85f365d286349:0x1da4e14975e45e72

HPI - Hasso-Plattner-Institut, Prof

HPI - Hasso-Plattner-Institut
5.0 ★★★★★ 20 reviews
Research Institute

Prof.-Dr.-Helmert-Straße 2-3, 14482 Potsdam
hpi.de
0331 55090
*Suggest an edit*

Rudolf-Breitscheid-Straße 196
14482 Potsdam
52.395868, 13.128933

Map data ©2016 GeoBasis-DE/BKG (©2009), Google

Software
Architecture Group

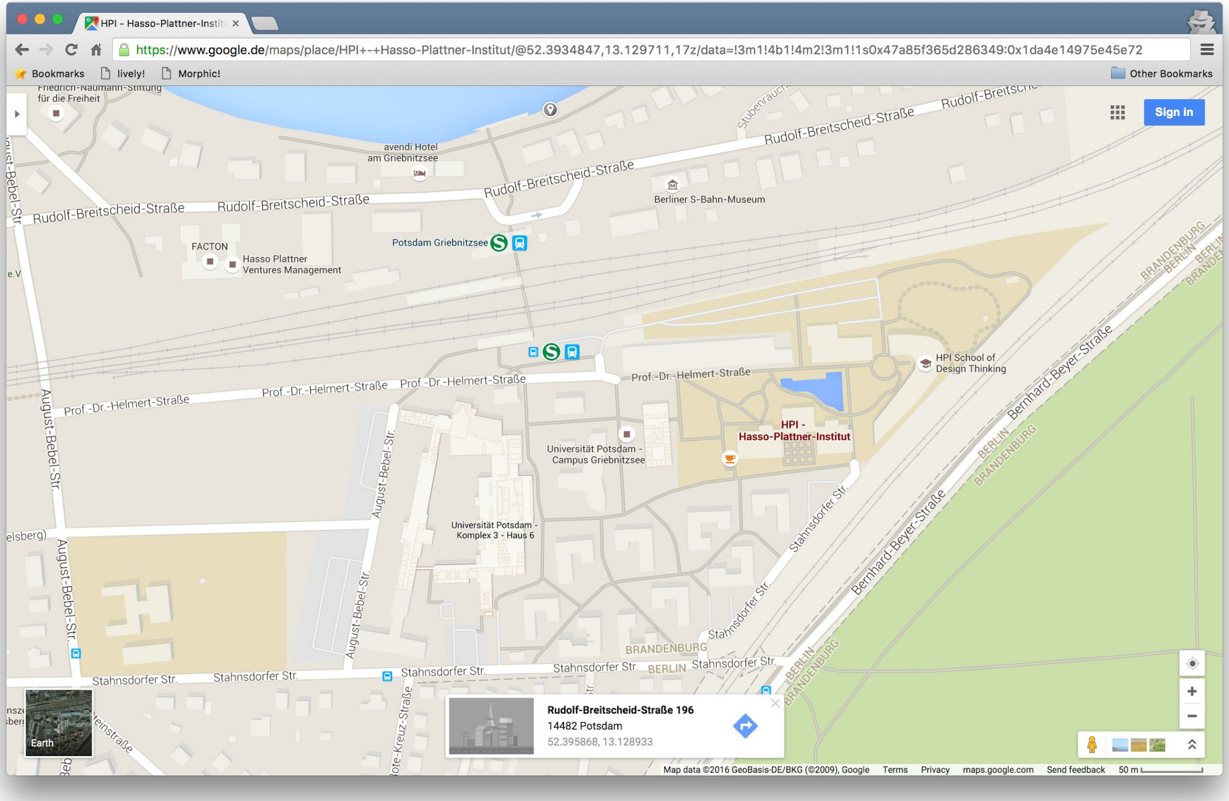
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Motivation: What we can have



HPI - Hasso-Plattner-Instit
https://www.google.de/maps/place/HPI+-+Hasso-Plattner-Institut/@52.3934847,13.129711,17z/data=!3m1!4b1!4m2!3m1!1s0x47a85f365d286349:0x1da4e14975e45e72
Bookmarks
lively!
Morphic!
Other Bookmarks
Sign in
Rudolf-Breitscheid-Straße
Prof.-Dr.-Helmert-Straße
August-Bebel-Str.
Stahnsdorfer Str.
Bernhard-Beyer-Straße
Potsdam Griebnitzsee
Berliner S-Bahn-Museum
avendi Hotel am Griebnitzsee
FACTON
Hasso Plattner Ventures Management
HPI School of Design Thinking
HPI - Hasso-Plattner-Institut
Universität Potsdam - Campus Griebnitzsee
Universität Potsdam - Komplex 3 - Haus 6
BRANDENBURG
BERLIN
Earth
Rudolf-Breitscheid-Straße 196
14482 Potsdam
52.395868, 13.128933
Map data ©2016 GeoBasis-DE/BKG (©2009), Google Terms Privacy maps.google.com Send feedback 50 m

Software Architecture Group

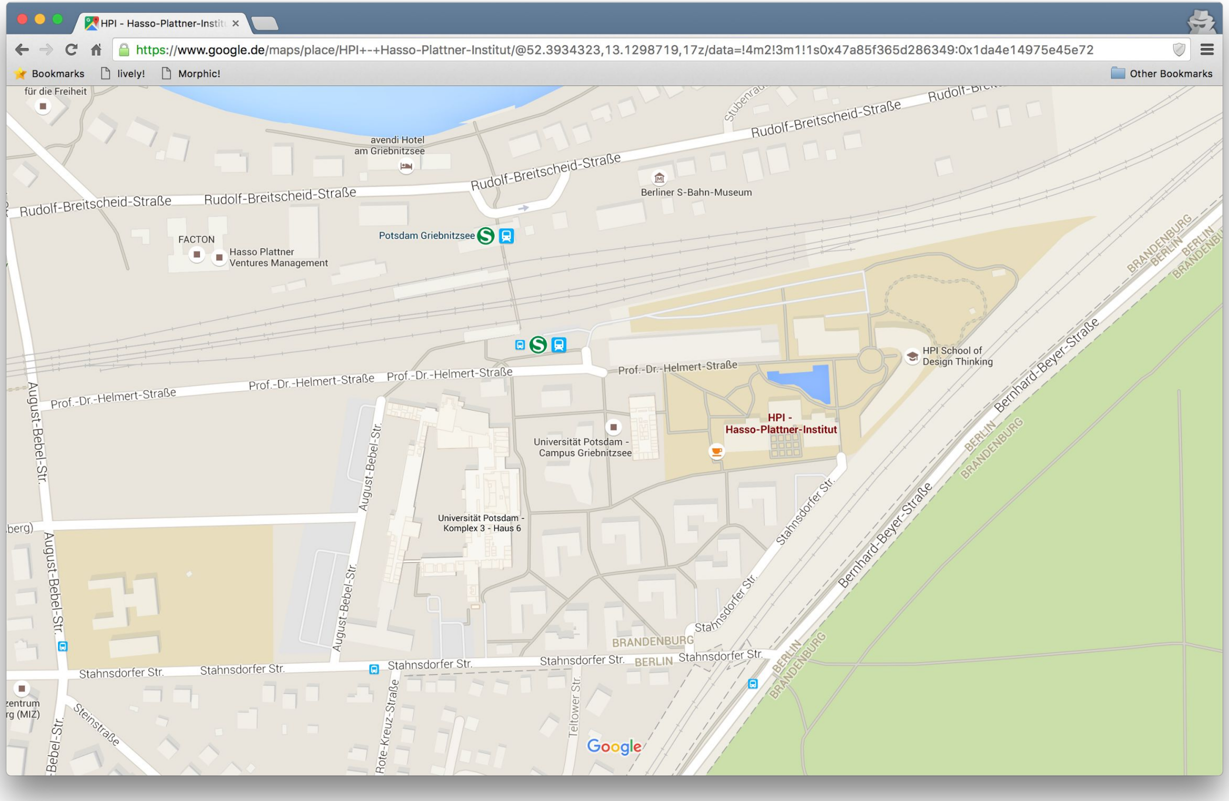
Software Design Seminar 2015/16

Topic: Lively4 Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Motivation: What we want to have


HPI - Hasso-Plattner-Instit
https://www.google.de/maps/place/HPI+-+Hasso-Plattner-Institut/@52.3934323,13.1298719,17z/data=!4m2!3m1!1s0x47a85f365d286349:0x1da4e14975e45e72
Bookmarks
lively!
Morphic!
Other Bookmarks
für die Freiheit
avendi Hotel am Griebnitzsee
Rudolf-Breitscheid-Straße
Berliner S-Bahn-Museum
FACTON
Hasso Plattner Ventures Management
Potsdam Griebnitzsee
Prof.-Dr.-Helmert-Straße
HPI School of Design Thinking
HPI - Hasso-Plattner-Institut
Universität Potsdam - Campus Griebnitzsee
Universität Potsdam - Komplex 3 - Haus 6
August-Bebel-Str.
Stahnsdorfer Str.
Bernhard-Beyer-Straße
BRANDENBURG
BERLIN
Rote-Kreuz-Straße
Teltower Str.
Steinstraße
zentrum rg (MIZ)
berg)
Google

Software
Architecture Group

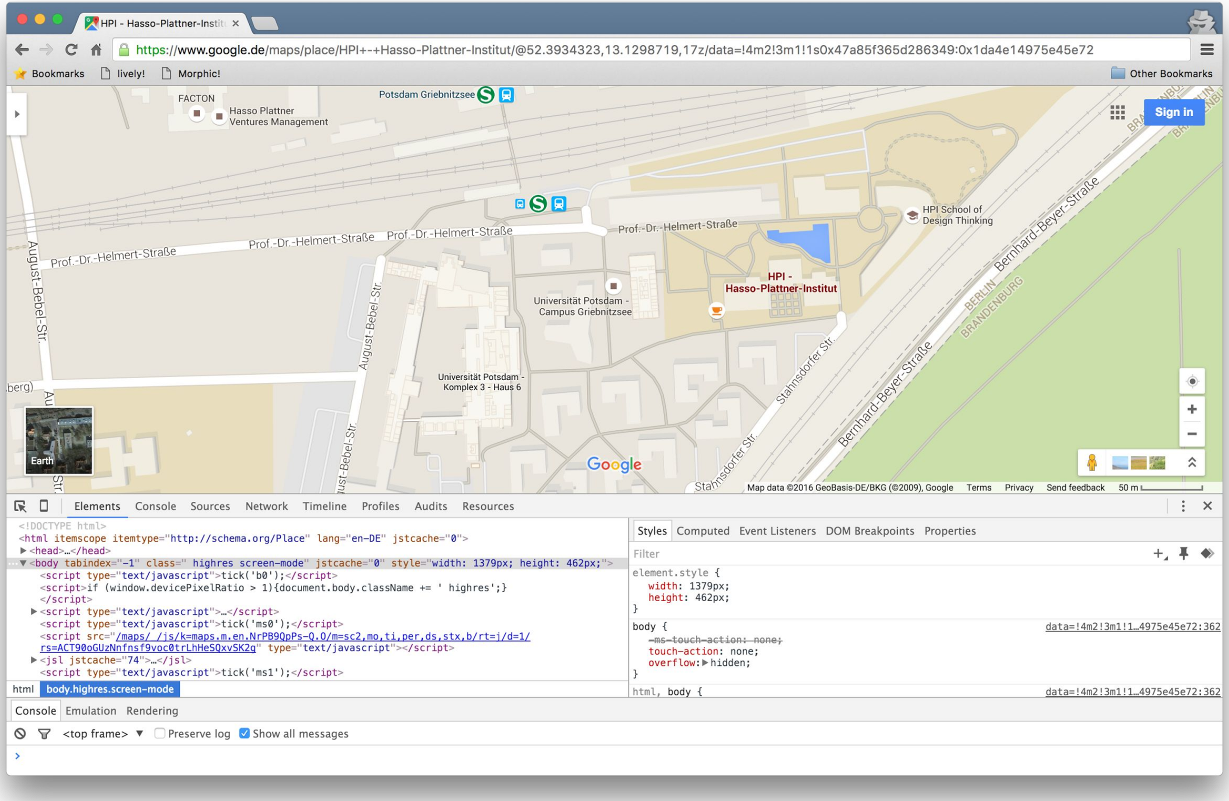
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Motivation: What we don’t want to do



```
<!DOCTYPE html>
<html itemscope itemtype="http://schema.org/Place" lang="en-DE" jstcache="0">
▶<head>…</head>
▼<body tabindex="-1" class=" highres screen-mode" jstcache="0" style="width: 1379px; height: 462px;">
   <script type="text/javascript">tick('b0');</script>
   <script>if (window.devicePixelRatio > 1){document.body.className += ' highres';}
   </script>
 ▶<script type="text/javascript">…</script>
   <script type="text/javascript">tick('ms0');</script>
   <script src="/maps/_/js/k=maps.m.en.NrPB9QpPs-Q.O/m=sc2,mo,ti,per,ds,stx,b/rt=j/d=1/
   rs=ACT90oGUzNnfnsf9voc0trLhHeSQxvSK2g" type="text/javascript"></script>
 ▶<jsl jstcache="74">…</jsl>
   <script type="text/javascript">tick('ms1');</script>
```

```
element.style {
   width: 1379px;
   height: 462px;
}
body {                                    data=!4m2!3m1!1…4975e45e72:362
   -ms-touch-action: none;
   touch-action: none;
   overflow:▶hidden;
}
html, body {                              data=!4m2!3m1!1…4975e45e72:362
```

Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Software Architecture Group

Software Design Seminar 2015/16

Topic: Lively4 Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Motivation

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

A

We want **direct manipulation** of **any page**

- Support basic morphic concepts
- Dragging, grabbing, deleting, ...
- Only operate on DOM structure

B

We want to build tools/widgets and **reuse** them in **any page**

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Background

Software
Architecture Group

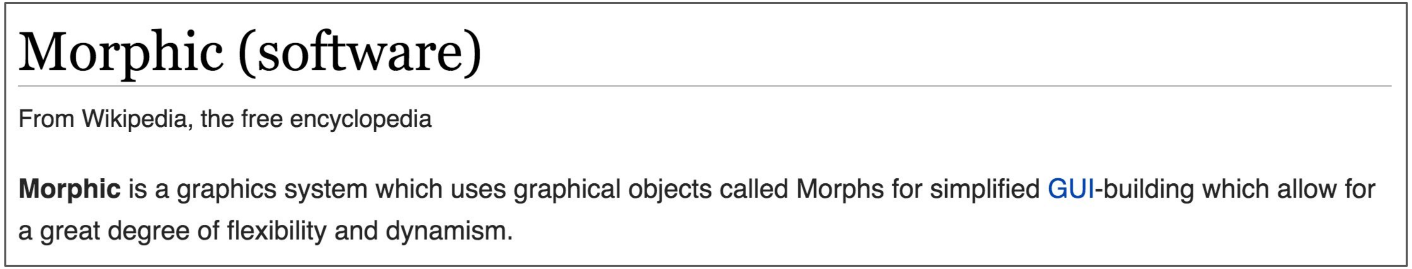
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Background



## Morphic (software)

From Wikipedia, the free encyclopedia

**Morphic** is a graphics system which uses graphical objects called Morphs for simplified GUI-building which allow for a great degree of flexibility and dynamism.

Typical elements of a morphic environment

- **Graphical objects**
- Direct manipulation
- **Morph composition**
- Inspection (state)
- Scripting (behavior)
- Connecting attributes (events / behavior)

**Thin Morphic**

Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Software
Architecture Group

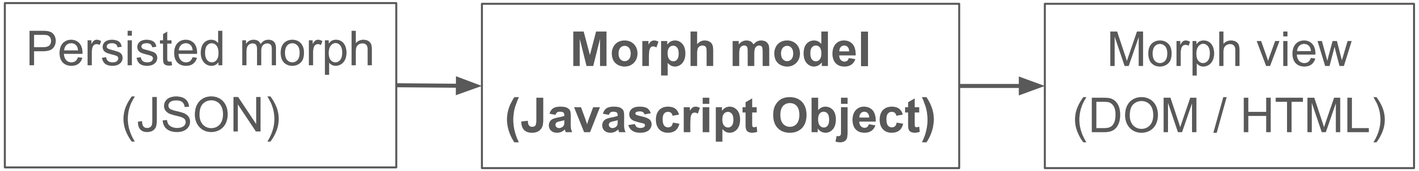
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Background: Morphic in Lively Kernel



Persisted morph (JSON) → **Morph model (Javascript Object)** → Morph view (DOM / HTML)

Morphs are represented by Javascript Objects

- **Independent** from rendering engine (HTML, SVG, canvas)
- Morph model is lively-specific, it is **bound to *lively-worlds***
  → Direct changes in HTML cannot affect morph model

Software
Architecture Group

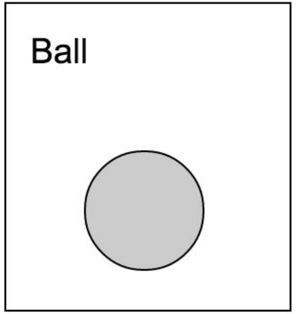
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

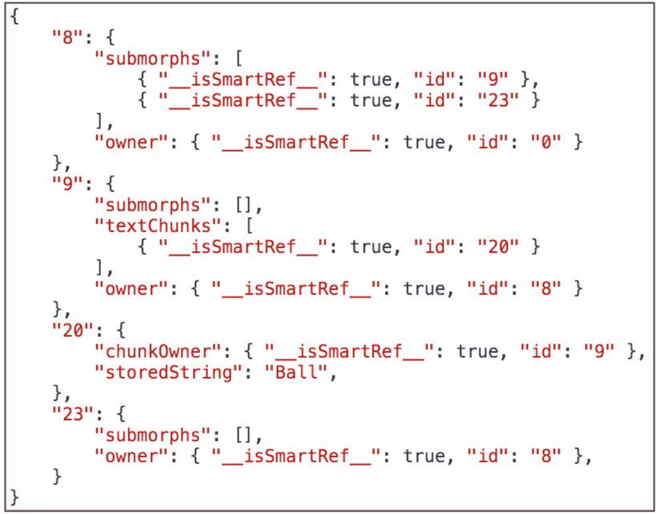
2016-02-02

Lixissimus
daniel-wer
athomschke

# Background: Lively Kernel morphic structure



Morph



```
{
    "8": {
        "submorphs": [
            { "__isSmartRef__": true, "id": "9" },
            { "__isSmartRef__": true, "id": "23" }
        ],
        "owner": { "__isSmartRef__": true, "id": "0" }
    },
    "9": {
        "submorphs": [],
        "textChunks": [
            { "__isSmartRef__": true, "id": "20" }
        ],
        "owner": { "__isSmartRef__": true, "id": "8" }
    },
    "20": {
        "chunkOwner": { "__isSmartRef__": true, "id": "9" },
        "storedString": "Ball",
    },
    "23": {
        "submorphs": [],
        "owner": { "__isSmartRef__": true, "id": "8" },
    }
}
```

JSON

Software
Architecture Group

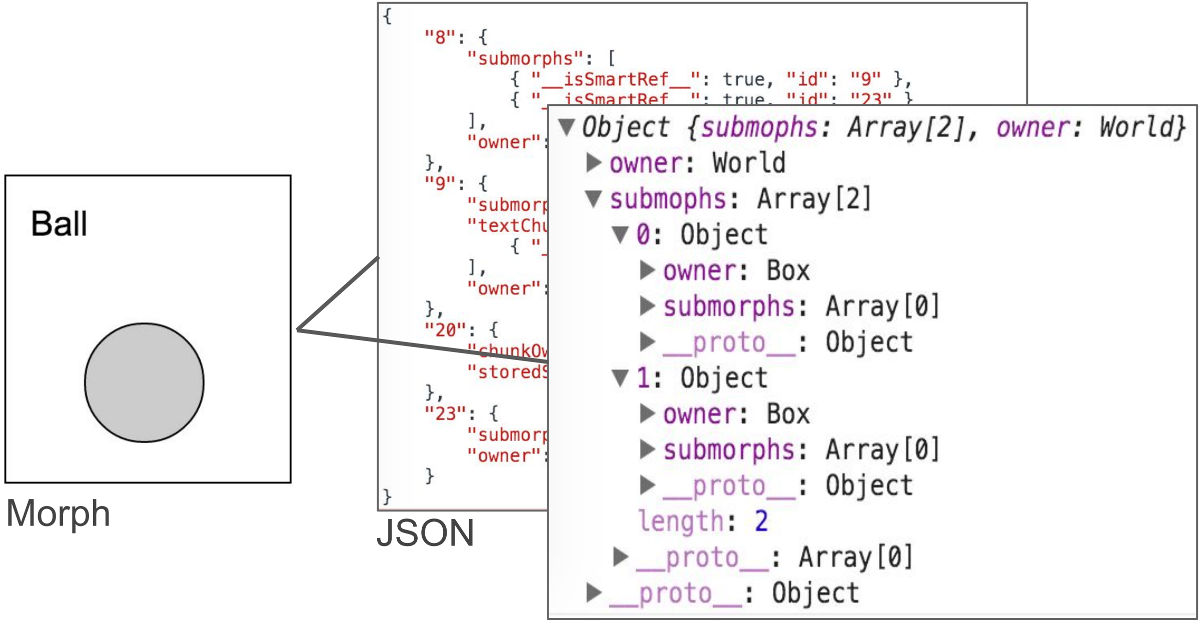
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Background: Lively Kernel morphic structure



Morph

```
{
    "8": {
        "submorphs": [
            { "__isSmartRef__": true, "id": "9" },
            { "__isSmartRef__": true, "id": "23" }
        ],
        "owner":
    },
    "9": {
        "submorp
        "textChu
            { "
        ],
        "owner":
    },
    "20": {
        "chunkOw
        "storedS
    },
    "23": {
        "submorp
        "owner":
    }
}
```

JSON

```
▼Object {submophs: Array[2], owner: World}
  ▶owner: World
  ▼submophs: Array[2]
    ▼0: Object
      ▶owner: Box
      ▶submorphs: Array[0]
      ▶__proto__: Object
    ▼1: Object
      ▶owner: Box
      ▶submorphs: Array[0]
      ▶__proto__: Object
     length: 2
    ▶__proto__: Array[0]
  ▶__proto__: Object
```

JS

Software
Architecture Group

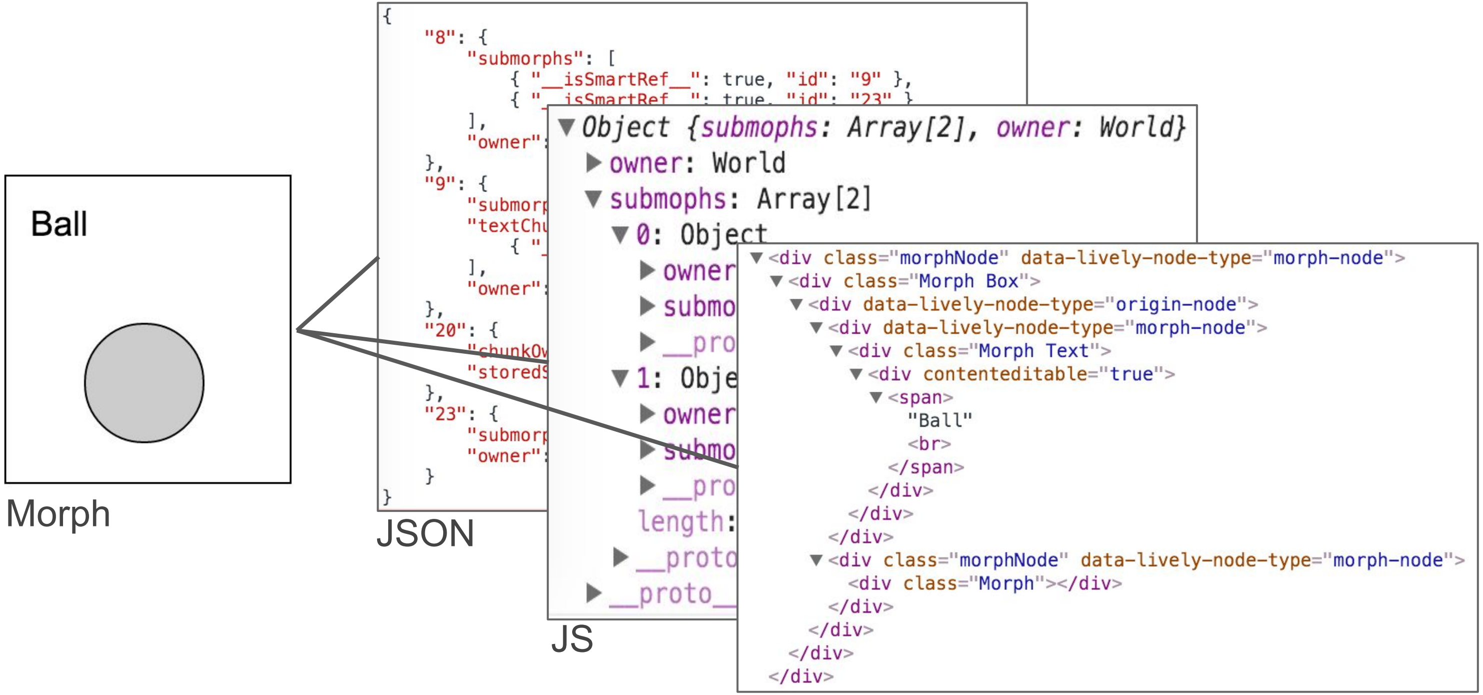
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Background: Lively Kernel morphic structure



```
{
    "8": {
        "submorphs": [
            { "__isSmartRef__": true, "id": "9" },
            { "__isSmartRef__": true, "id": "23" }
        ],
        "owner":
    },
    "9": {
        "submorp
        "textChu
            { "_
        ],
        "owner":
    },
    "20": {
        "chunkOw
        "storedS
    },
    "23": {
        "submorp
        "owner":
    }
}
```

```
▼Object {submophs: Array[2], owner: World}
  ▶owner: World
  ▼submophs: Array[2]
    ▼0: Object
      ▶owner
      ▶submo
      ▶__pro
    ▼1: Obje
      ▶owner
      ▶submo
      ▶__pro
      length:
    ▶__proto
  ▶__proto_
```

```
▼<div class="morphNode" data-lively-node-type="morph-node">
  ▼<div class="Morph Box">
    ▼<div data-lively-node-type="origin-node">
      ▼<div data-lively-node-type="morph-node">
        ▼<div class="Morph Text">
          ▼<div contenteditable="true">
            ▼<span>
                "Ball"
                <br>
              </span>
            </div>
          </div>
        </div>
      ▼<div class="morphNode" data-lively-node-type="morph-node">
          <div class="Morph"></div>
        </div>
      </div>
    </div>
  </div>
```

Morph

JSON

JS

DOM

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

IT Systems Engineering | Universität Potsdam

# Approach

Software
Architecture Group

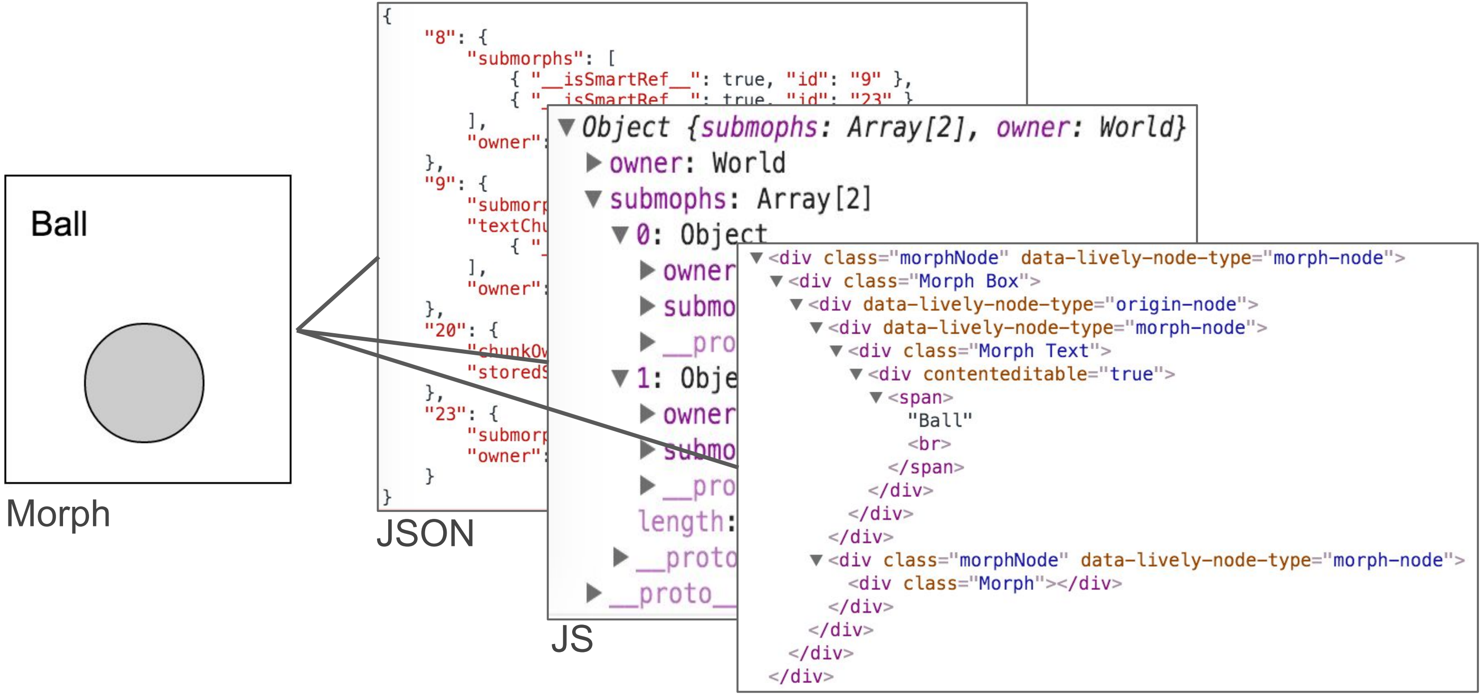
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Approach: Lively4 morphic structure



```
{
    "8": {
        "submorphs": [
            { "__isSmartRef__": true, "id": "9" },
            { "__isSmartRef__": true, "id": "23" }
        ],
        "owner":
    },
    "9": {
        "submorp
        "textChu
            { "
        ],
        "owner":
    },
    "20": {
        "chunkOw
        "storedS
    },
    "23": {
        "submorp
        "owner":
    }
}
```

```
▼Object {submophs: Array[2], owner: World}
  ▶owner: World
  ▼submophs: Array[2]
    ▼0: Object
      ▶owner
      ▶submo
      ▶__pro
    ▼1: Obje
      ▶owner
      ▶submo
      ▶__pro
      length:
    ▶__proto
  ▶__proto_
```

```
▼<div class="morphNode" data-lively-node-type="morph-node">
  ▼<div class="Morph Box">
    ▼<div data-lively-node-type="origin-node">
      ▼<div data-lively-node-type="morph-node">
        ▼<div class="Morph Text">
          ▼<div contenteditable="true">
            ▼<span>
                "Ball"
                <br>
              </span>
            </div>
          </div>
        </div>
      ▼<div class="morphNode" data-lively-node-type="morph-node">
          <div class="Morph"></div>
        </div>
      </div>
    </div>
  </div>
</div>
```

Morph

JSON

JS

DOM

Software
Architecture Group

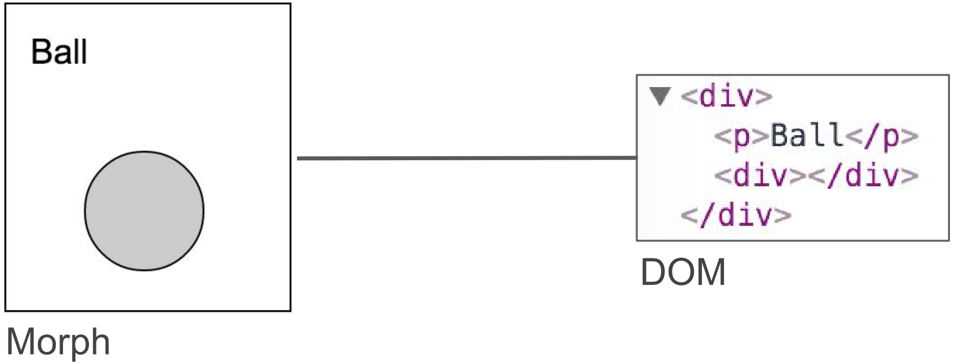
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Approach: Lively4 morphic structure



```
▼ <div>
    <p>Ball</p>
    <div></div>
  </div>
```

Software
Architecture Group

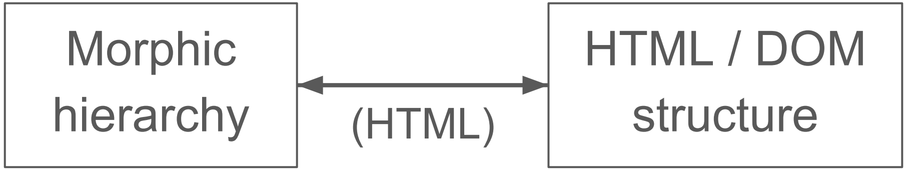
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Approach: Morphs in Lively4



Morphs **are** DOM nodes

- Single point of truth (HTML)
- Every website is morphic-ready
  → Morphic tools work out of the box

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

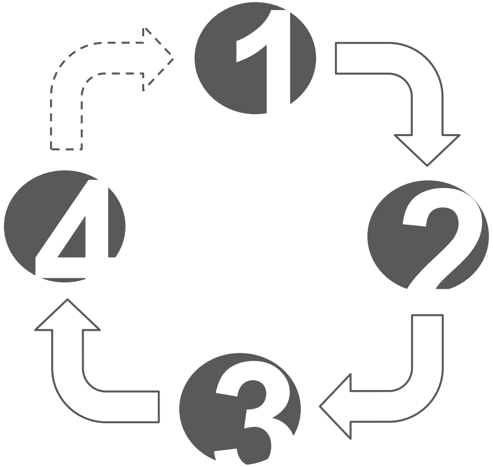
Lixissimus
daniel-wer
athomschke

# Workflow

# Workflow

**Build**

- Interactively manipulate web pages
- Create tools



**Export**

- Persist as template
- Hide inner structure

**Use**

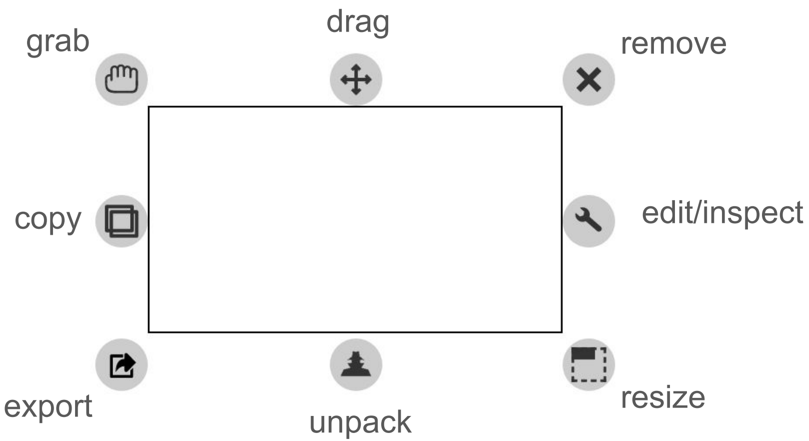
- Import in other pages
- Automatically load dependencies

**Improve**

- Unpack custom element from shadow DOM

# Build: Interactively change DOM with Halo

- Standard HTML elements as rudimentary building blocks (e.g. <div>)
- Use halo to interactively change and compose morphs



Software Architecture Group

Software Design Seminar 2015/16

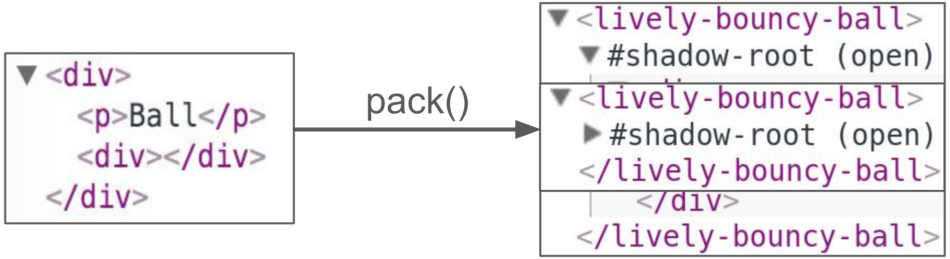
Topic: Lively4 Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Export: Reusable components

Custom HTML elements for more complex morphs
→ Allowed by Web Components specification



```
▼<div>
   <p>Ball</p>
   <div></div>
 </div>
```

pack()

```
▼<lively-bouncy-ball>
  ▼#shadow-root (open)
▼<lively-bouncy-ball>
  ▶#shadow-root (open)
 </lively-bouncy-ball>
    </div>
 </lively-bouncy-ball>
```

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Web Component specification

- **Custom Elements:** *readable dom structure*
- **Shadow Dom:** *hide complexity, encapsulate style*
- **HTML Templates:** *define components*
- **HTML Imports:** *include and reuse templates*

Software
Architecture Group

Software Design
Seminar 2015/16

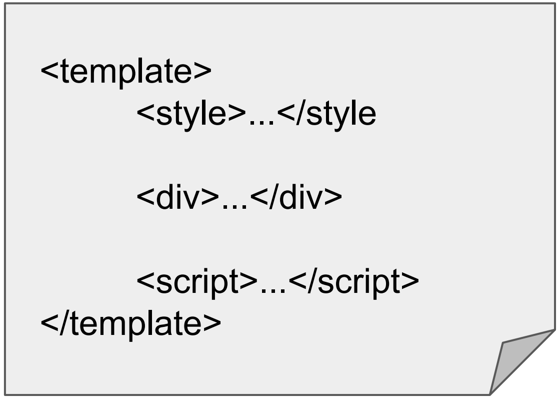
Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Export: Web Component Templates

- Saved in HTML file
- Self contained (style, structure, behaviour)
- Can be imported in arbitrary web pages via HTML imports
- Loaded dynamically and asynchronously (does not block js thread)



```
<template>
        <style>...</style

        <div>...</div>

        <script>...</script>
</template>
```

Software
Architecture Group

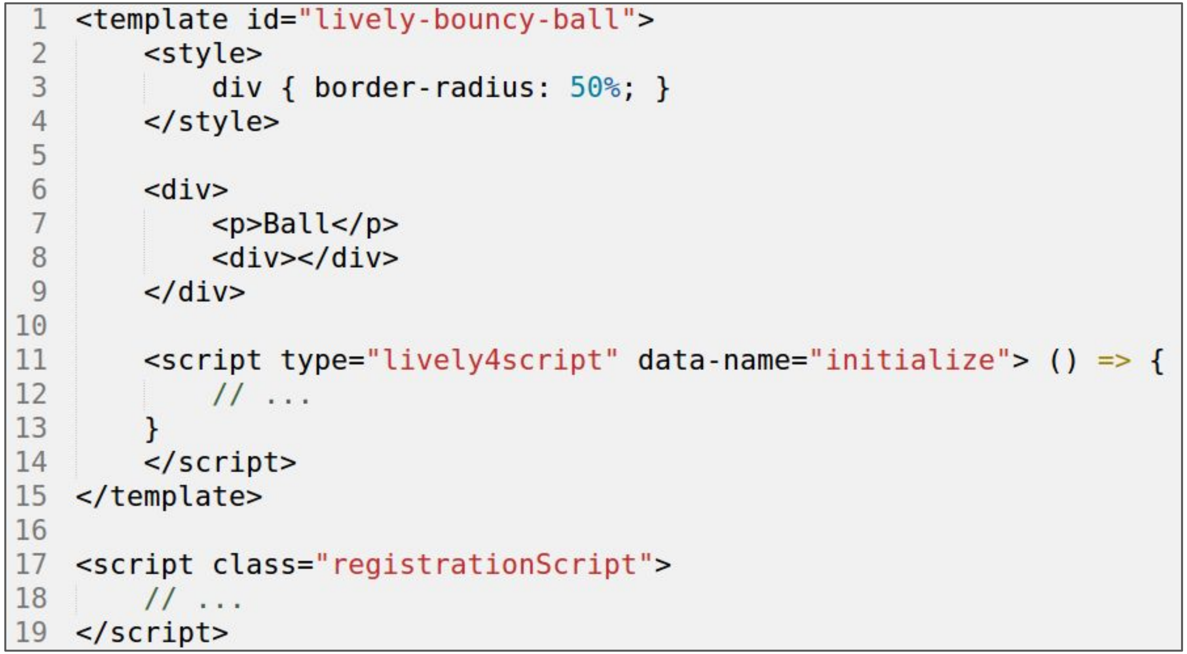
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Export: lively-test-component.html



```
<template id="lively-bouncy-ball">
    <style>
        div { border-radius: 50%; }
    </style>

    <div>
        <p>Ball</p>
        <div></div>
    </div>

    <script type="lively4script" data-name="initialize"> () => {
        // ...
    }
    </script>
</template>

<script class="registrationScript">
    // ...
</script>
```

Software
Architecture Group

Software Design
Seminar 2015/16

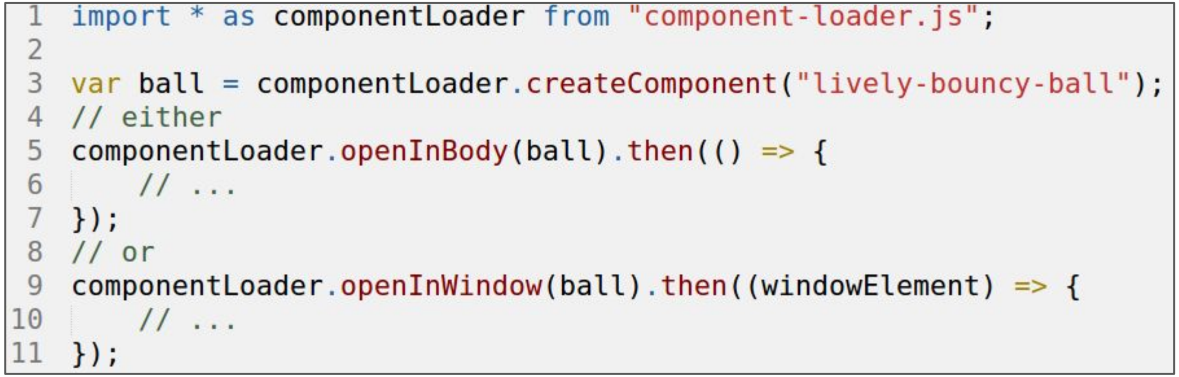
Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Use: Component Loading Example

Components can be created using the componentLoader API



```
import * as componentLoader from "component-loader.js";

var ball = componentLoader.createComponent("lively-bouncy-ball");
// either
componentLoader.openInBody(ball).then(() => {
    // ...
});
// or
componentLoader.openInWindow(ball).then((windowElement) => {
    // ...
});
```

Software
Architecture Group

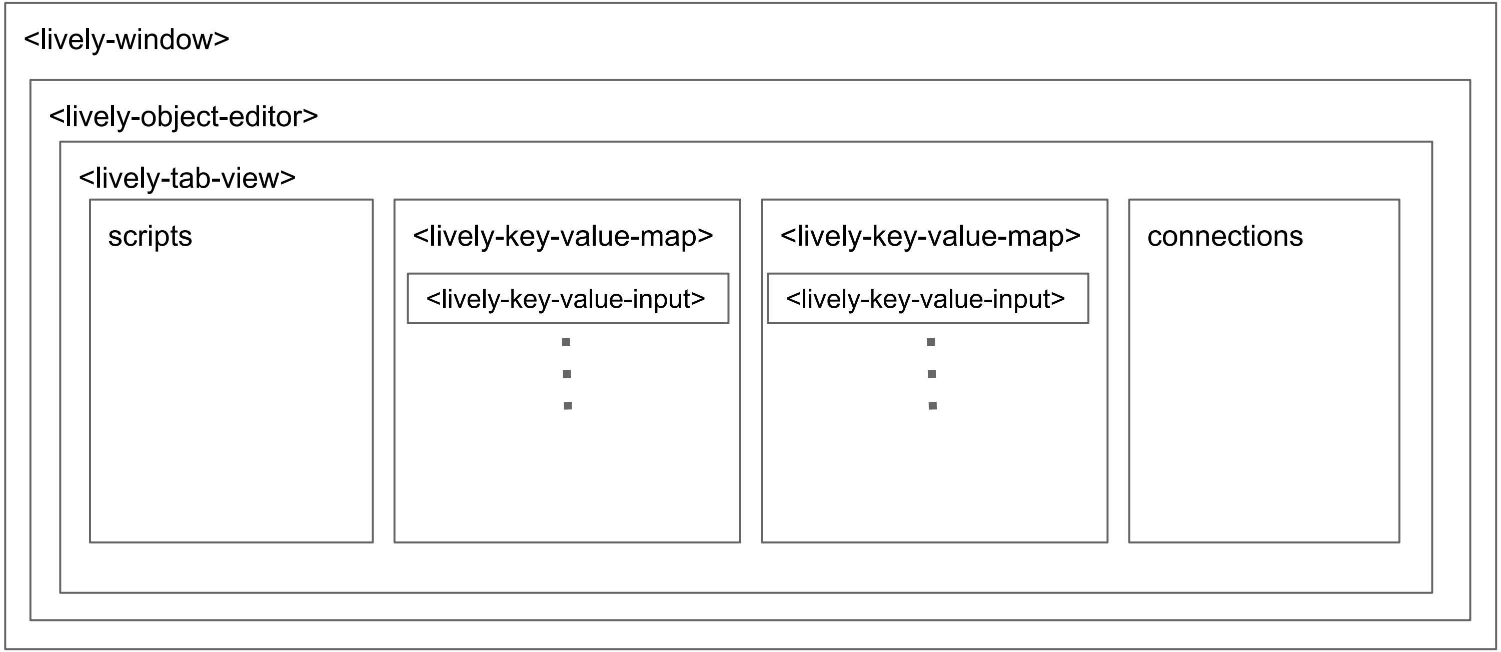
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

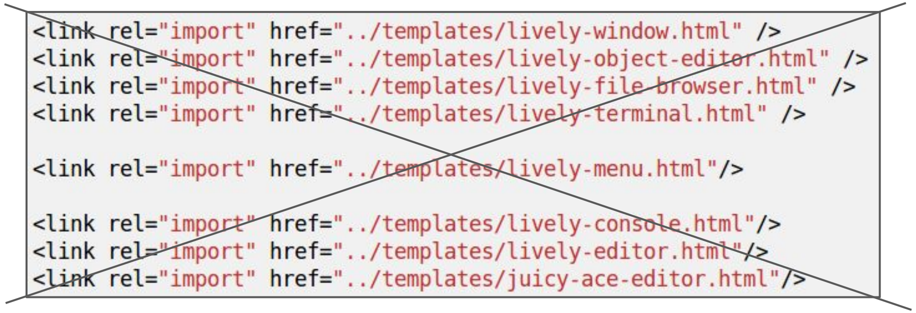
Lixissimus
daniel-wer
athomschke

# Use: Component Loading Dependencies



# Use: Component Loading Dependencies

- Loading of nested components
  - Subcomponents are loaded recursively
  - Call to initialize is deferred until all subcomponents are loaded properly
- Templates for unresolved tags are imported automatically
- No import statements needed in component definitions
  - No need to manage and clean up import tags
  - No need to persist import tags



```
<link rel="import" href="../templates/lively-window.html" />
<link rel="import" href="../templates/lively-object-editor.html" />
<link rel="import" href="../templates/lively-file-browser.html" />
<link rel="import" href="../templates/lively-terminal.html" />

<link rel="import" href="../templates/lively-menu.html"/>

<link rel="import" href="../templates/lively-console.html"/>
<link rel="import" href="../templates/lively-editor.html"/>
<link rel="import" href="../templates/juicy-ace-editor.html"/>
```

IT Systems Engineering | Universität Potsdam

Software
Architecture Group

Software Design
Seminar 2015/16

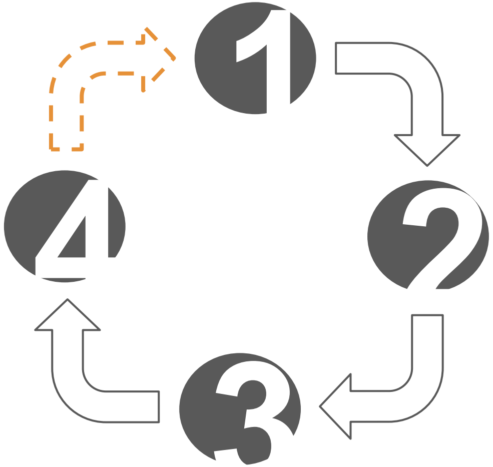
Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Demo

# Workflow



**Build**

- Interactively manipulate web pages
- Create tools

**Export**

- Persist as template
- Hide inner structure

**Use**

- Import in other pages
- Automatically load dependencies

**Improve**

- Unpack custom element from shadow DOM

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Evaluation & Future Work

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Evaluation: Advantages

- Open System
  - External Web Components can be easily imported and used
  - Web Components developed in Lively can be easily exported and used everywhere
- Lively benefits from further development of web standards
- Lower entry barrier than Lively Kernel
  → HTML & Web Components are well documented

# Evaluation: Problems

- **New parent element after exporting**
- Redundant boilerplate code for component registration in every template
  → But it allows loading into other pages
- Collecting style rules during export
  → What about dynamically attached style classes?

Software
Architecture Group

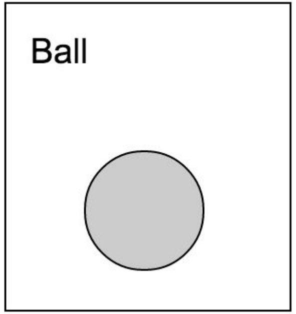
Software Design
Seminar 2015/16

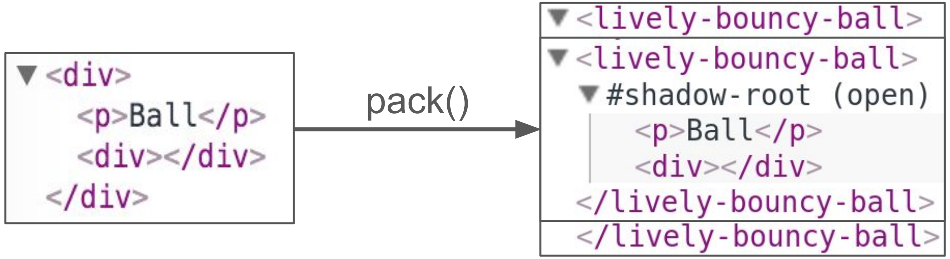
Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Evaluation: New Parent Element





```
▼<div>
   <p>Ball</p>
   <div></div>
 </div>
```

```
▼<lively-bouncy-ball>
▼<lively-bouncy-ball>
  ▼#shadow-root (open)
      <p>Ball</p>
      <div></div>
 </lively-bouncy-ball>
 </lively-bouncy-ball>
```

Software
Architecture Group

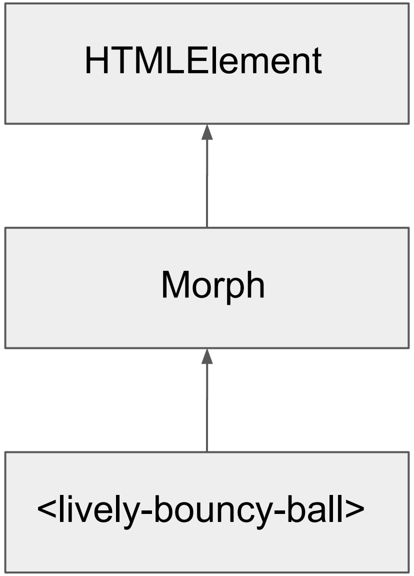
Software Design
Seminar 2015/16

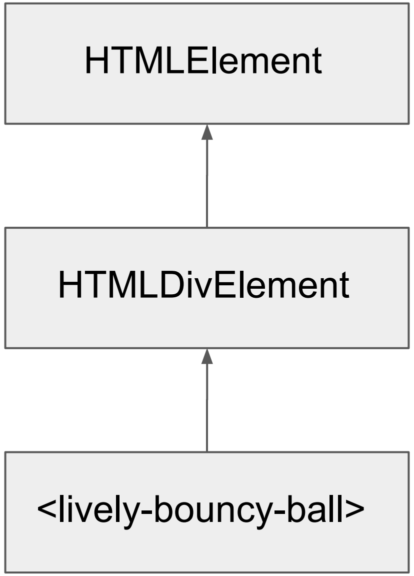
Topic: Lively4
Thin Morphic

2016-02-02

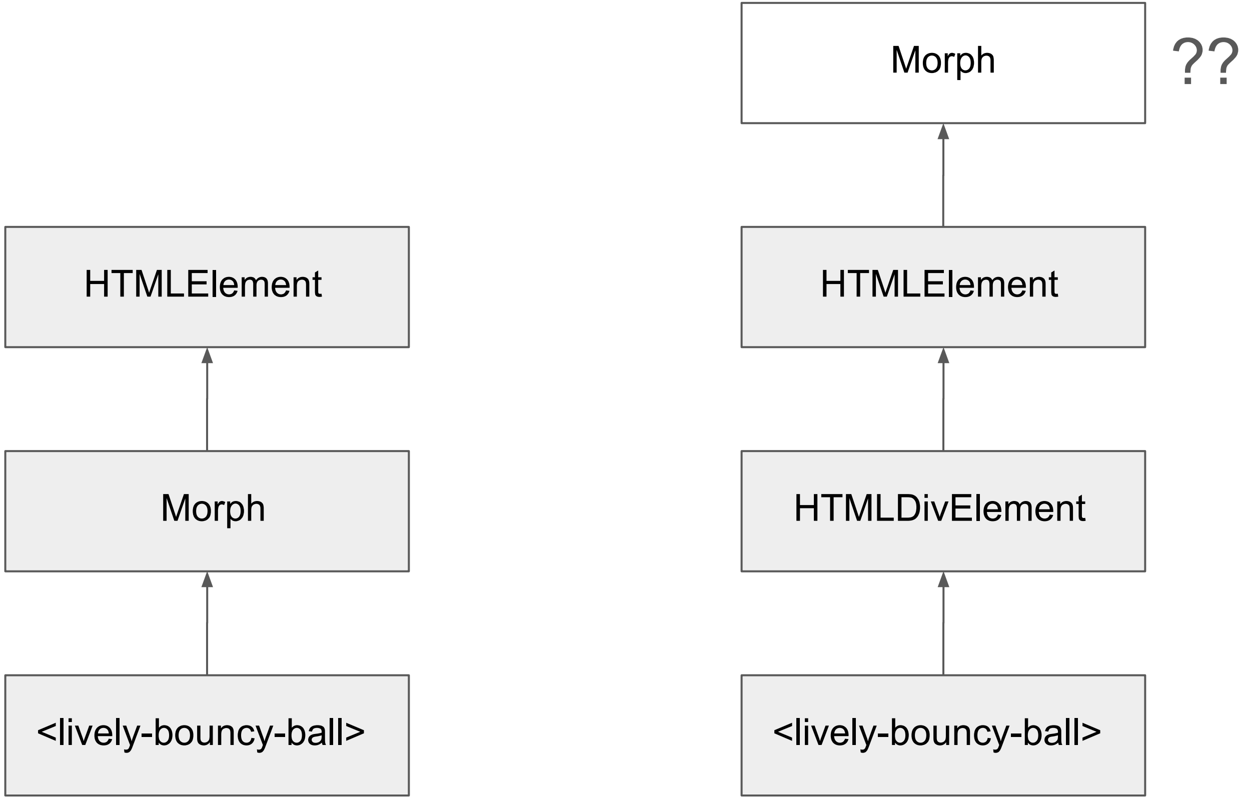
Lixissimus
daniel-wer
athomschke

# Evaluation: New Parent Element (cont.)





# Evaluation: New Parent Element (cont.)



# Future Work

Work for current seminar

- Refactor Chrome Extension

Work for future seminars

- Persistence in other pages
- Complete web component development cycle
  - re-publish after unpacking shadow DOM
  - interactive creation/styling of components (style editor)
- Component inheritance & versioning

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Summary

A We can directly manipulate any page.

B We can build tools and use them on any page.
(build, export, use, improve)

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Lively4: Thin Morphic

Software Design Seminar 2015/2016
Software Architecture Group

Felix Wolff
Daniel Werner
Astrid Thomschke

# Outtakes

Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Software
Architecture Group

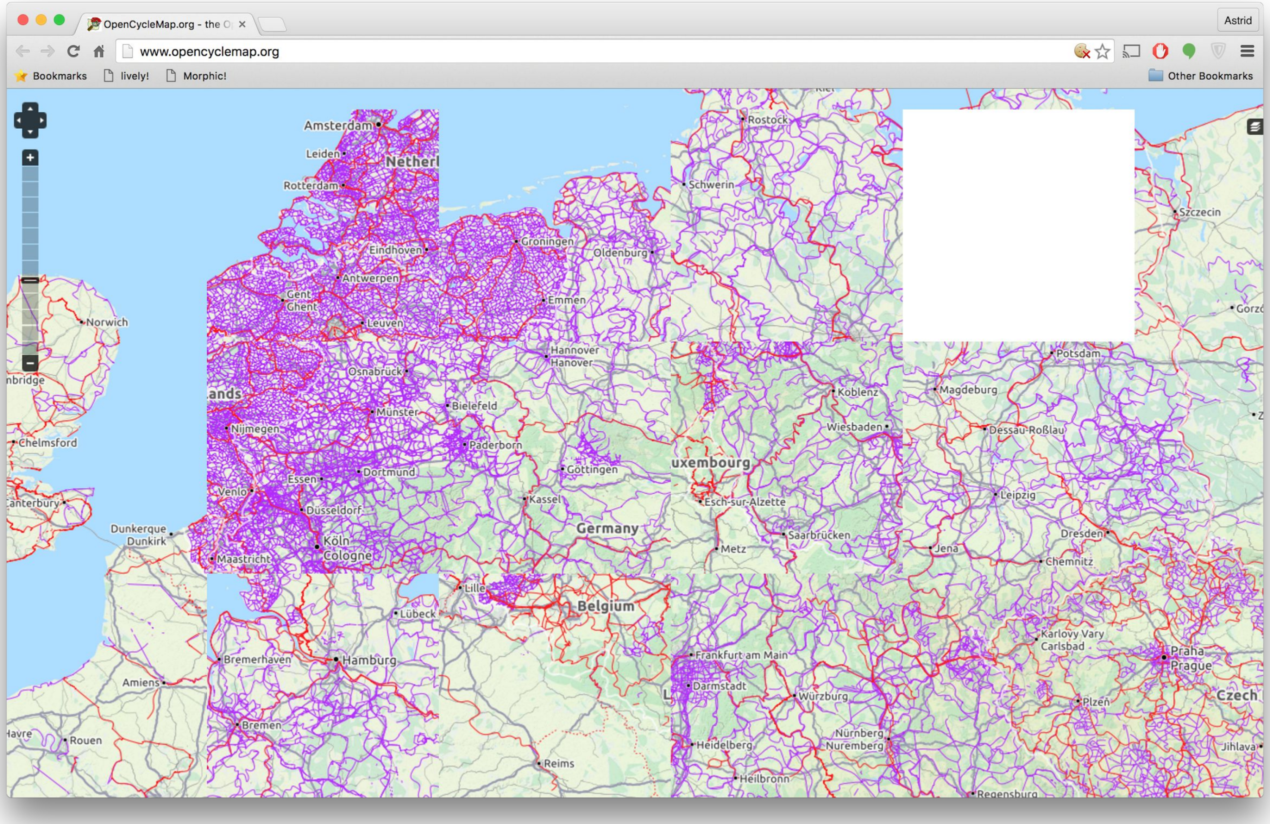
Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Opencyclemap



IT Systems Engineering | Universität Potsdam

Software Architecture Group

Software Design Seminar 2015/16

Topic: Lively4 Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke

# Backup

Software
Architecture Group

Software Design
Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

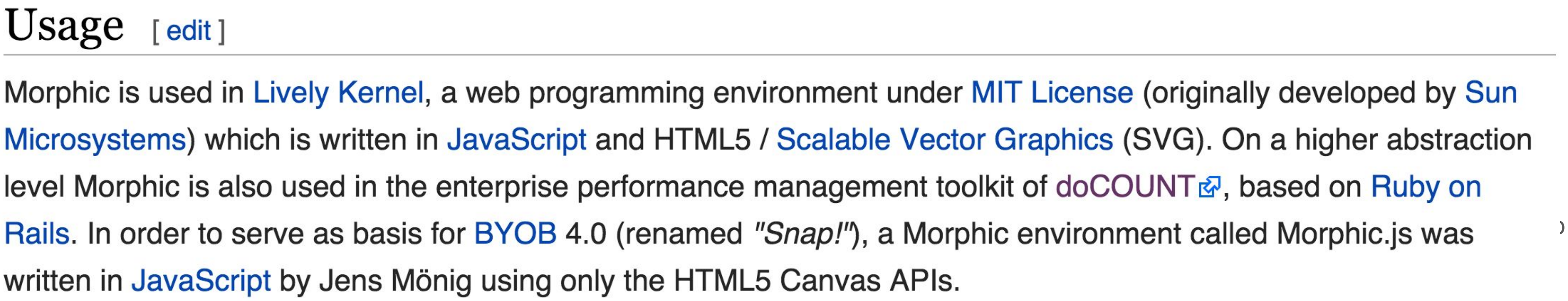
Lixissimus
daniel-wer
athomschke

# Future Question

How should Lively4 be used?

- As a library?
- As a “System”?

# Background: Morphic implementations



## Usage [ edit ]

Morphic is used in Lively Kernel, a web programming environment under MIT License (originally developed by Sun Microsystems) which is written in JavaScript and HTML5 / Scalable Vector Graphics (SVG). On a higher abstraction level Morphic is also used in the enterprise performance management toolkit of doCOUNT, based on Ruby on Rails. In order to serve as basis for BYOB 4.0 (renamed *"Snap!"*), a Morphic environment called Morphic.js was written in JavaScript by Jens Mönig using only the HTML5 Canvas APIs.

Morphic implementations:

- Lively Kernel
- doCOUNT (‘on a higher abstraction level’)
- morphic.js (canvas only)

Seminar 2015/16

Topic: Lively4
Thin Morphic

2016-02-02

Lixissimus
daniel-wer
athomschke