Structural Design Patterns

Patterns and Antipatterns in Javascript

Chrysa Papadaki // chr.papadaki@tum.de

Nishant Gupta // nishant.gupta@tum.de

Master Seminar | Technische Universität München | Garching, 29.09.2015

Outline

- ❏ Introduction
- ❏ Structural Design Patterns
- ❏ Applied Structural Patterns
- ❏ Proxy
- ❏ Decorator
- ❏ Project 1: Songfinder
- ❏ Facade
- ❏ Composite
- ❏ Project 2: drawr-bootstrap
- ❏ Q&A

Introduction

The categories of object-oriented patterns formed by Gang of Four (GoF):

Creational patterns are used to create objects

Structural patterns are used to combine objects and classes in order to build structured objects

Behavioral patterns are used to build a computation and control data flows

Structural Design Patterns

- ★ Form larger structures from individual parts
- ★ Vary a great deal depending on what sort of structure is being created for what purpose
- ★ Use inheritance to compose interfaces or implementations

Structural Design Patterns (cont.)

- ❏ Adapter
- ❏ Bridge
- ❏ Composite
- ❏ Decorator
- ❏ Facade
- ❏ Flyweight
- ❏ Proxy

Applied Structural Patterns

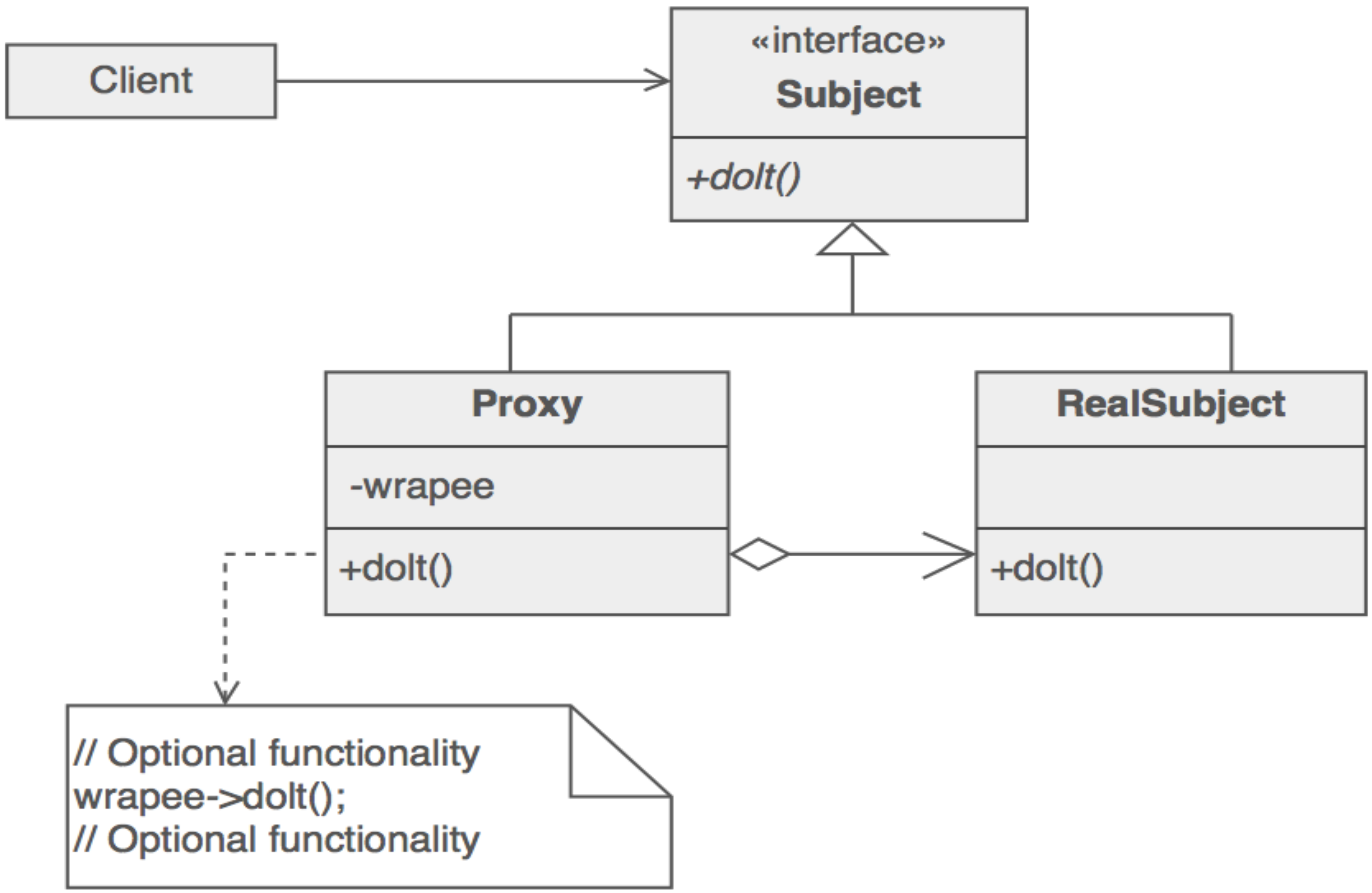
- ❑ Adapter
- ❑ Bridge
- ❑ **Composite**
- ❑ **Decorator**
- ❑ **Facade**
- ❑ Flyweight
- ❑ **Proxy**

Proxy Design Pattern

Proxy Pattern - Definition

- ★ Provides a surrogate for another object to control access to it
- ★ Uses an extra level of indirection to support distributed, controlled, or intelligent access
- ★ Adds a wrapper to protect the real component from undue complexity

Proxy Pattern - Structure



Proxy Pattern - Applicability

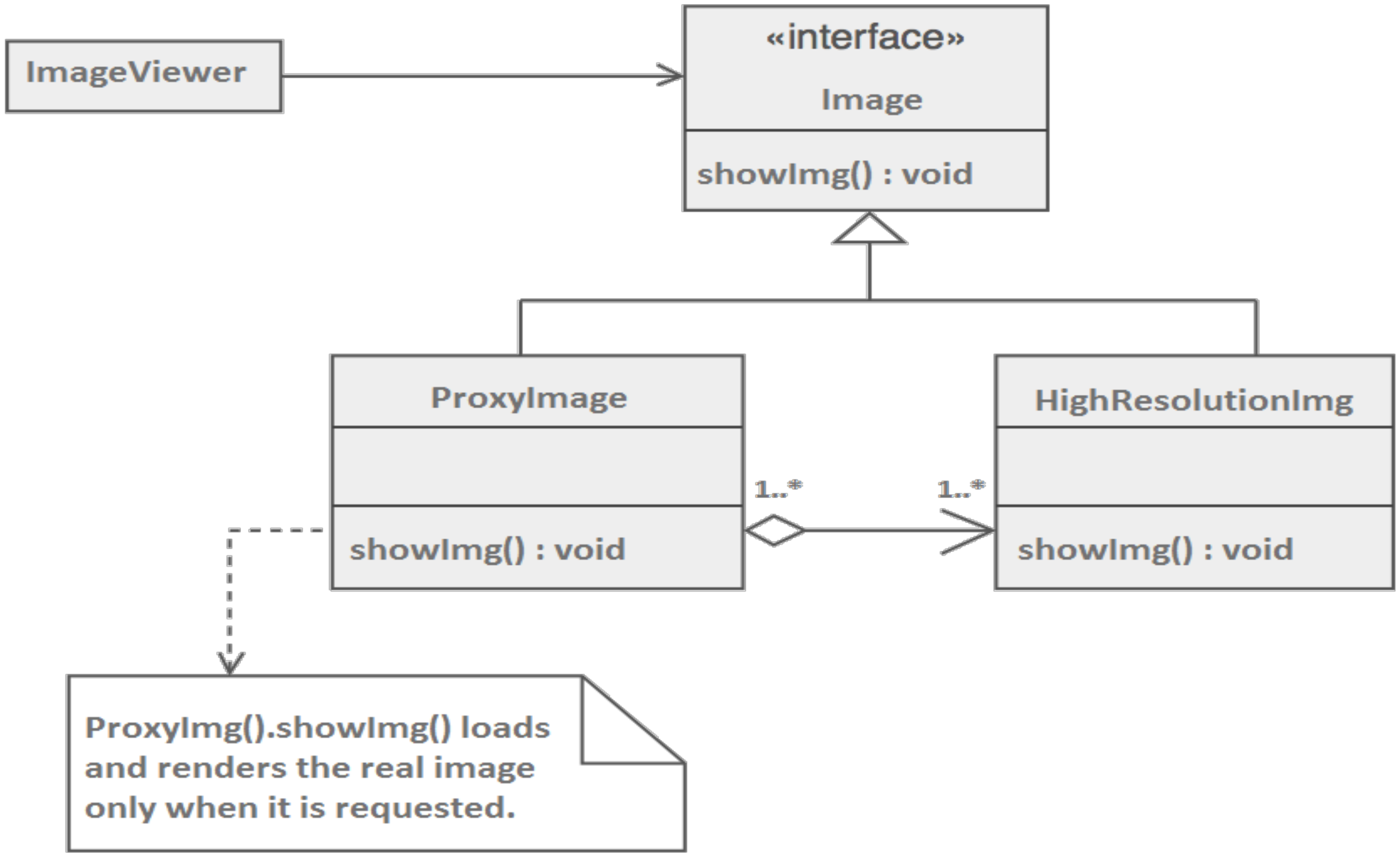
Virtual Proxies delay initialization of expensive objects

Remote Proxies represent locally a remote object

Protection Proxies control access to a sensitive object

Smart References interpose additional actions when an object is accessed

Proxy Pattern - Example



Proxy Pattern - Trade-off

Less efficiency due to indirection

Complex implementation

Decorator Design Pattern

Decorator Pattern - Definition

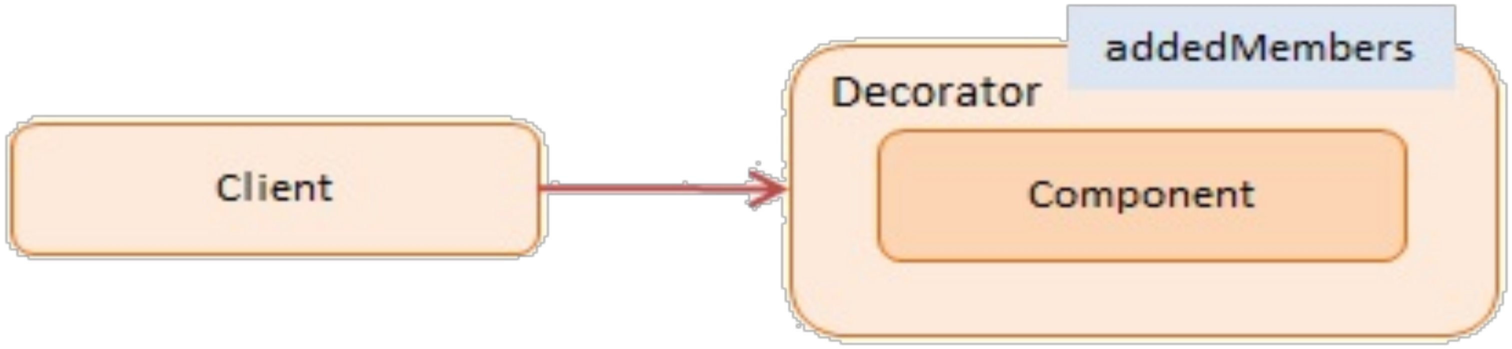
★ Attaches additional functionality dynamically

★ An alternative to subclassing for extending functionality

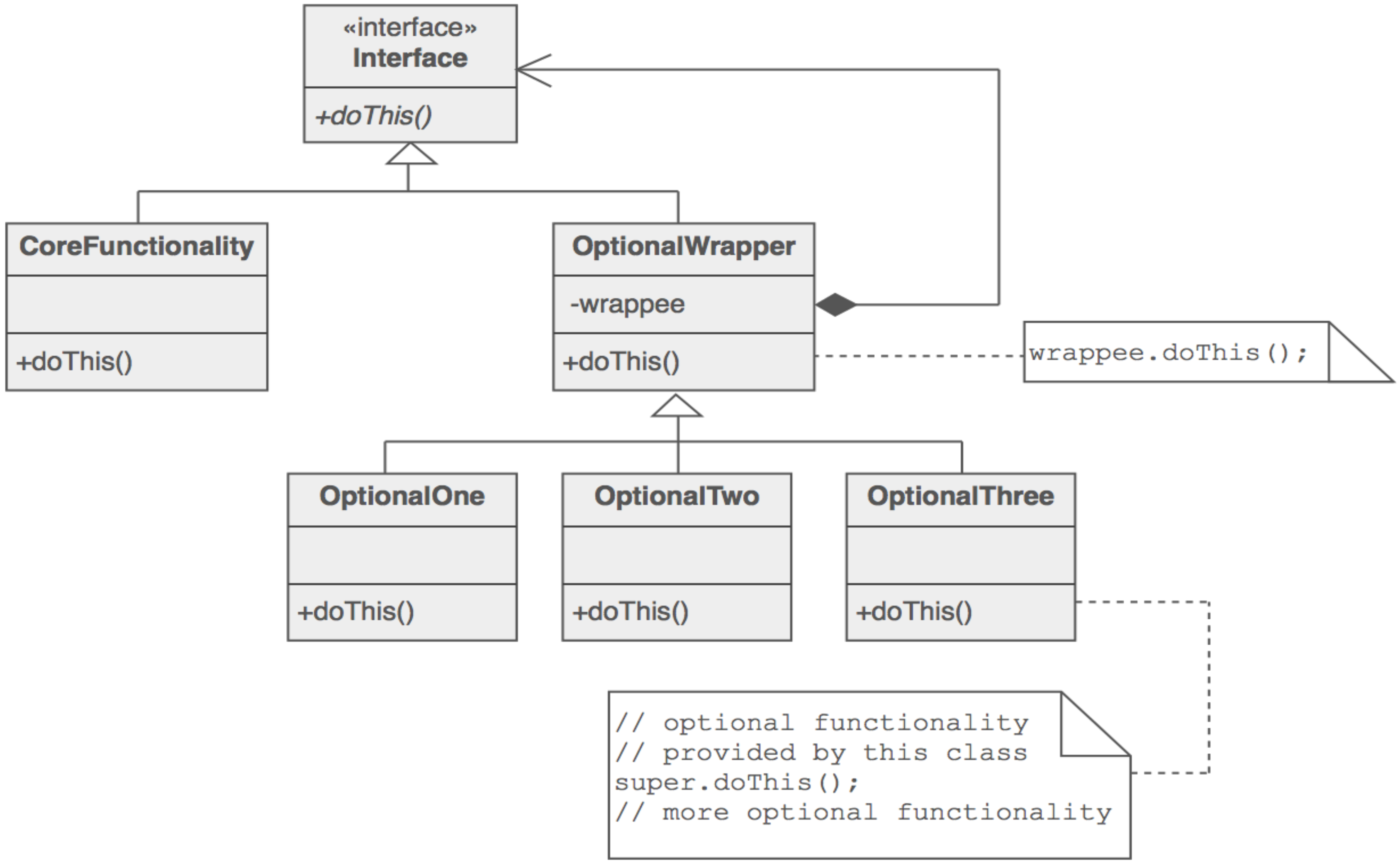
★ Encourages code reuse

★ A.k.a. **Wrapper**

Decorator Pattern - High Level



Decorator Pattern - Structure



```
wrappee.doThis();
```

```
// optional functionality
// provided by this class
super.doThis();
// more optional functionality
```

Decorator Pattern - Example

```
var User = function(name) {
    this.name = name;

    this.say = function() {
        log.add("User: " + this.name);
    };
}
```

A User object is going to be enhanced by a DecoratedUser object

Decorator Pattern - Example (cont)

```
var DecoratedUser = function(user, street, city) {
    this.user = user;
    this.name = user.name; // ensures interface stays the same
    this.street = street;
    this.city = city;

    this.say = function() {
        log.add("Decorated User: " + this.name + ", " +
                    this.street + ", " + this.city);
    };
}
```

Decorator Pattern - Example (cont)

```
function run() {
    var user = new User("Andreas");
    user.say();

    var decorated = new DecoratedUser(user, "Bolzstr", "Ulm");
    decorated.say();

    log.show();
}
```

```
console >
User: Andreas
Decorated User: Andreas, Bolzstr, Ulm
```

Decorator Pattern - Trade-off

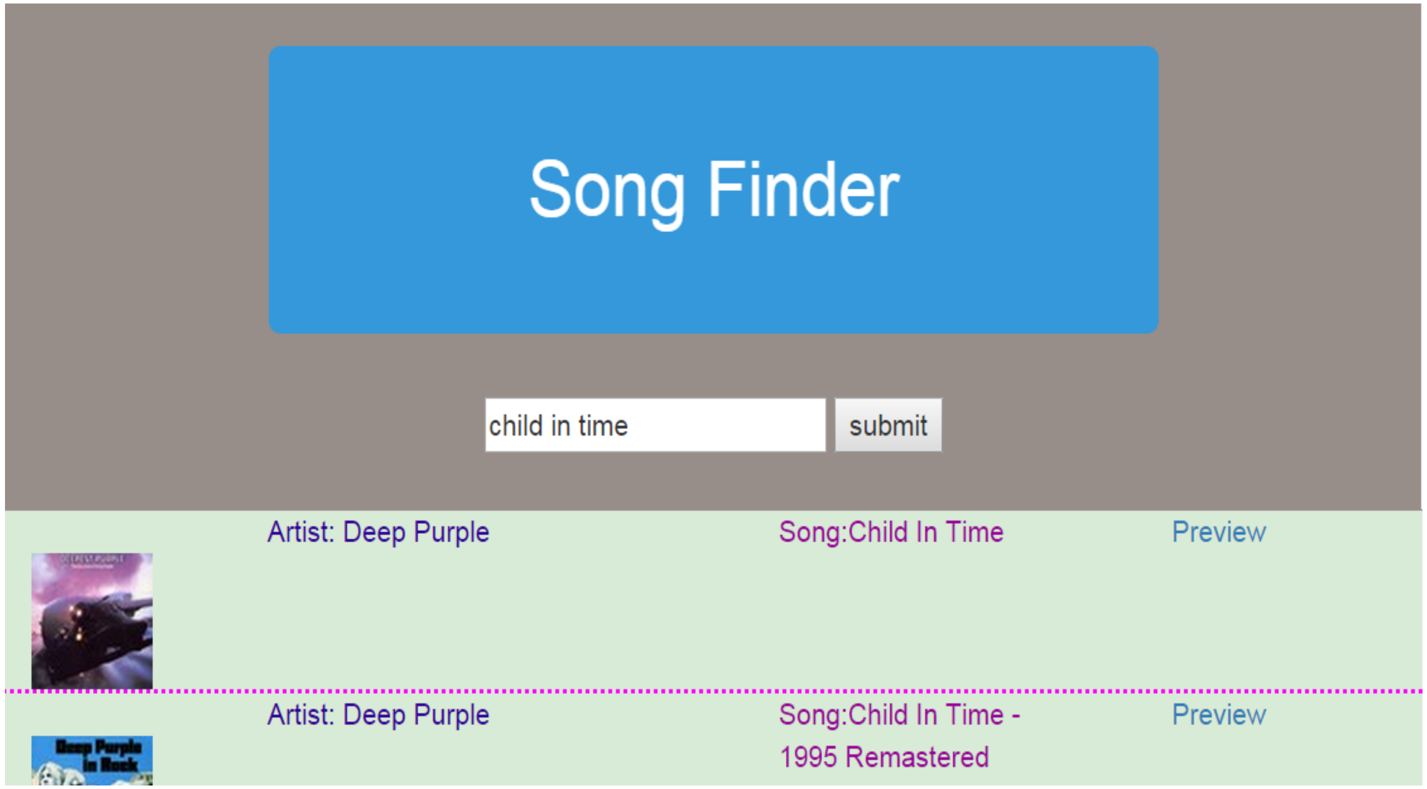
Decorators can result in many small objects

Overuse can be complex

Project: songFinder

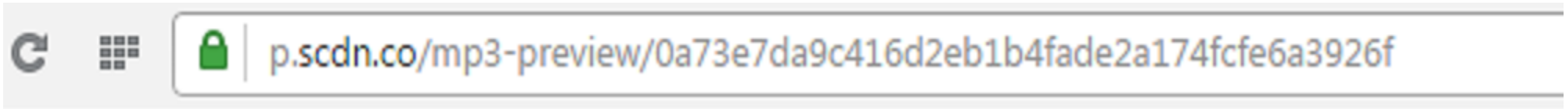
songFinder

Song search engine using spotify web services

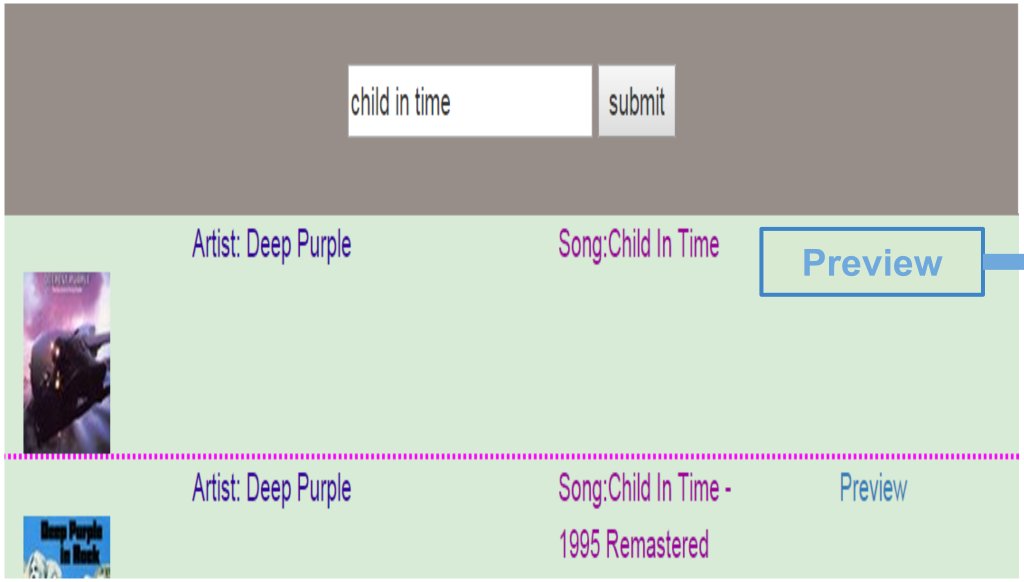


	Artist: Deep Purple	Song:Child In Time	Preview
	Artist: Deep Purple	Song:Child In Time - 1995 Remastered	Preview

songFinder - Song Preview



p.scdn.co/mp3-preview/0a73e7da9c416d2eb1b4fade2a174fcfe6a3926f

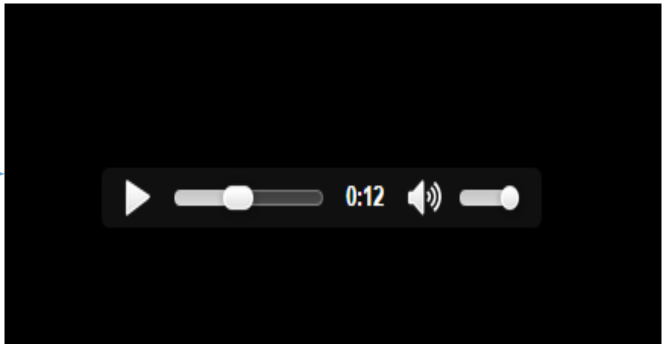


child in time

submit

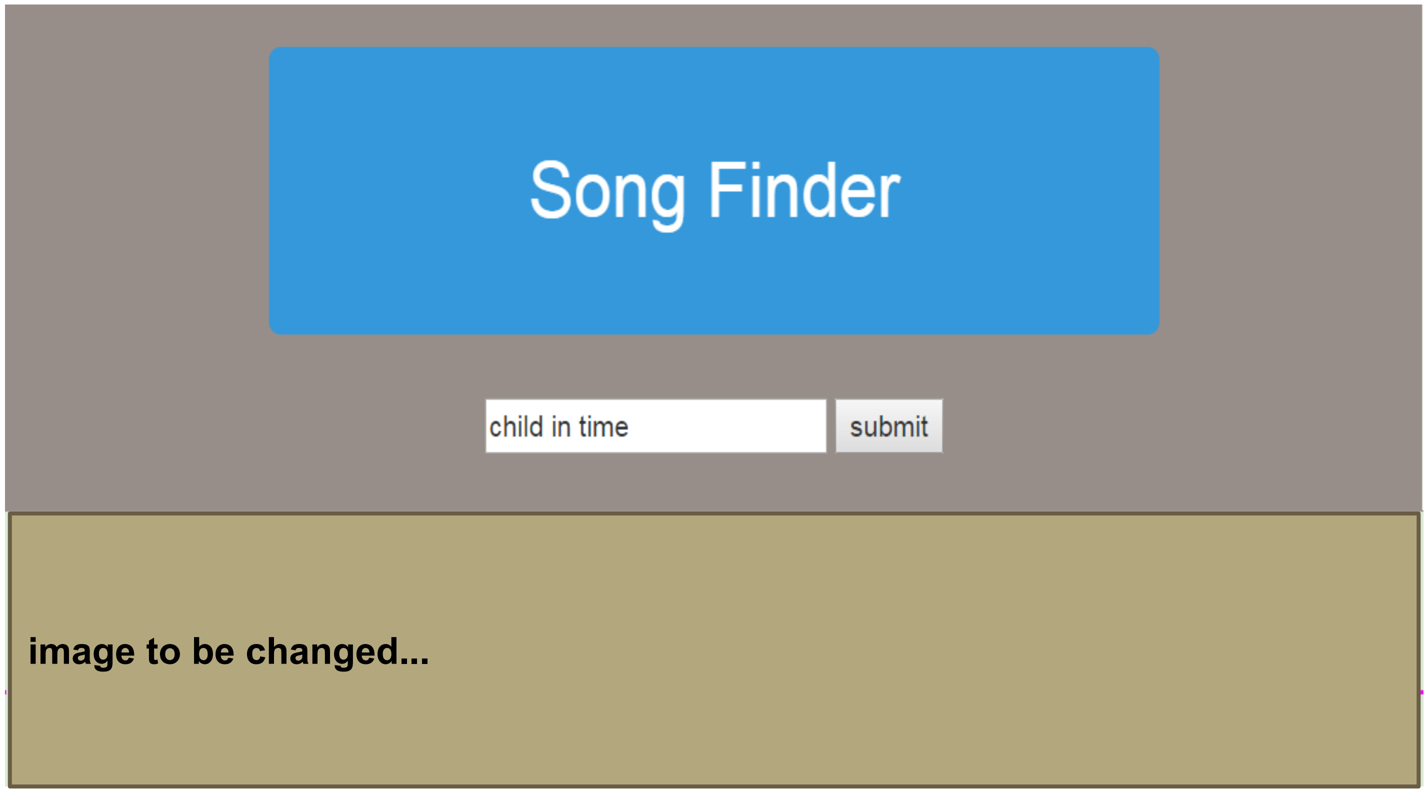
Artist: Deep Purple Song:Child In Time Preview

Artist: Deep Purple Song:Child In Time - 1995 Remastered Preview



songFinder - Single Result

On single result the song plays automatically - no need to press Preview



songFinder - Symptoms

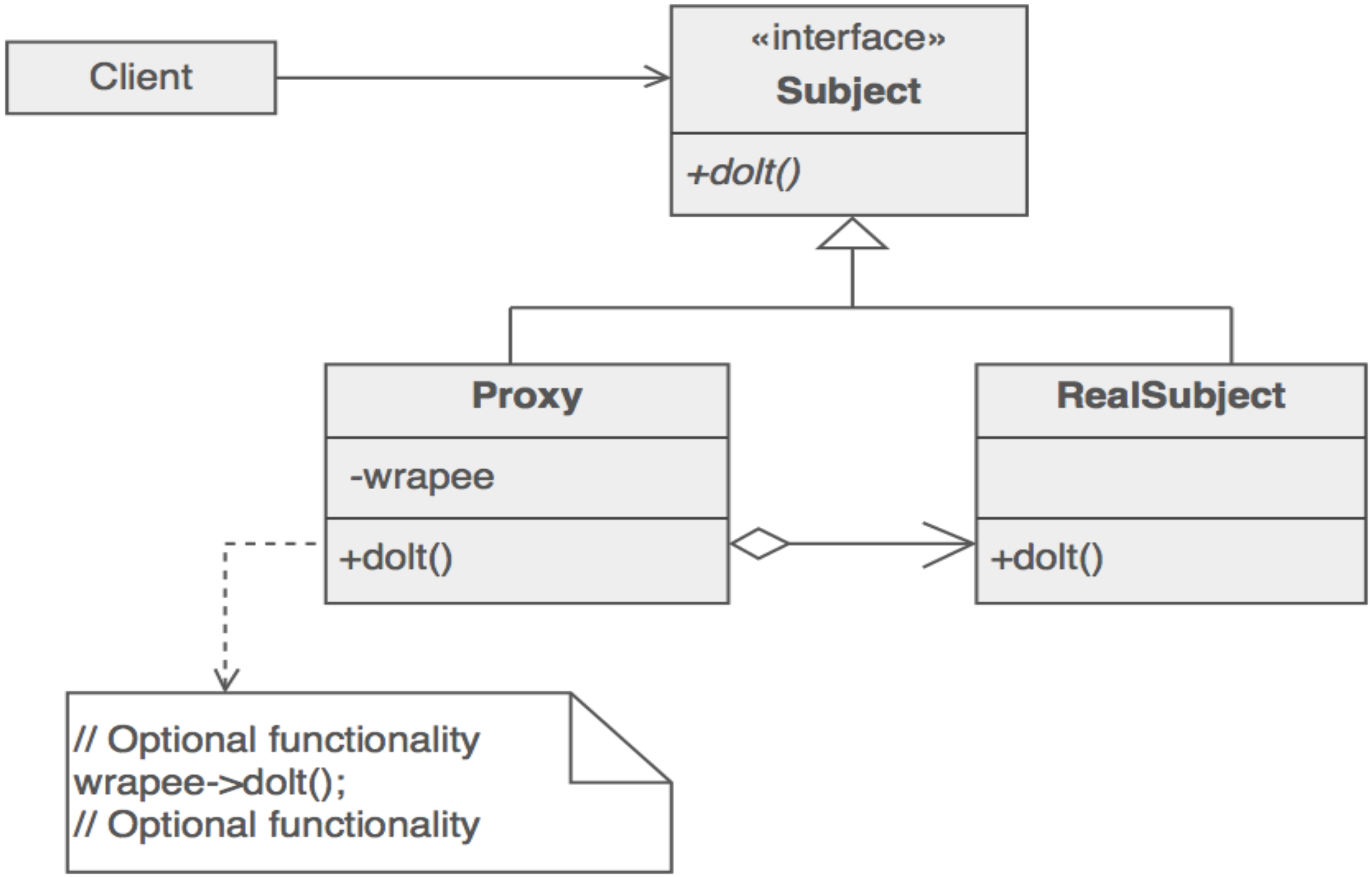
No caching: everytime long list of songs is being fetched and rendered

Tight coupling: parsing and rendering the ws response in one place

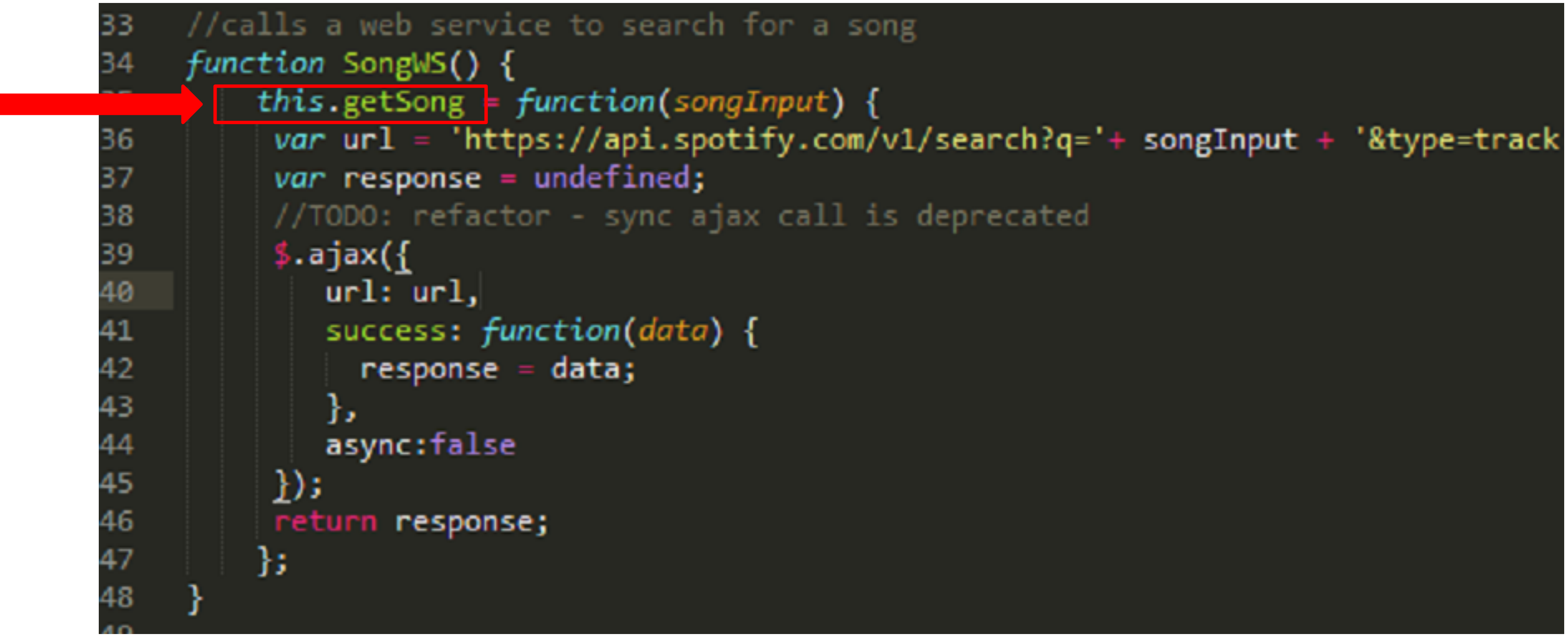
Code Smells:

```
// display(createSongsObj(getSongArray(data), Song) );
```

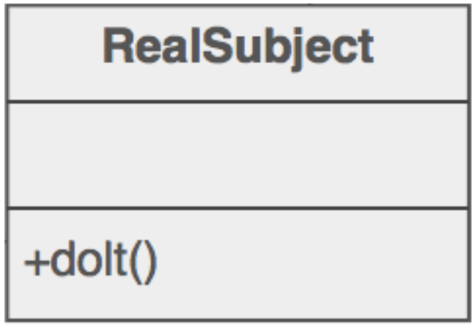
songFinder - Refactoring using Proxy



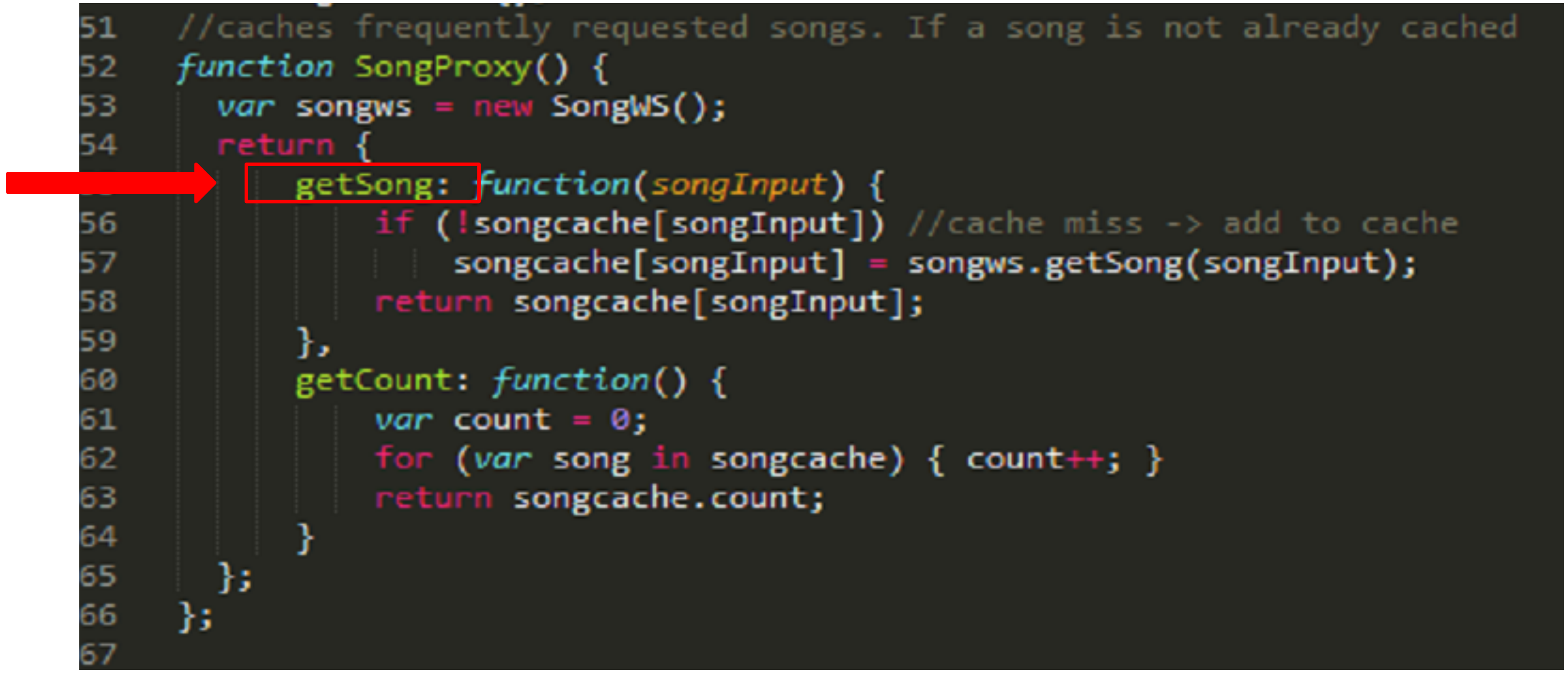
songFinder - Refactoring using Proxy



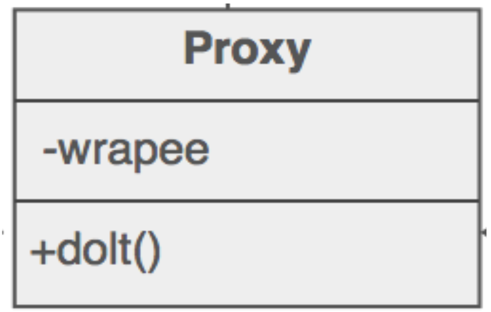
```
//calls a web service to search for a song
function SongWS() {
    this.getSong = function(songInput) {
        var url = 'https://api.spotify.com/v1/search?q='+ songInput + '&type=track
        var response = undefined;
        //TODO: refactor - sync ajax call is deprecated
        $.ajax({
            url: url,
            success: function(data) {
                response = data;
            },
            async:false
        });
        return response;
    };
}
```



songFinder - Refactoring using Proxy (cont)

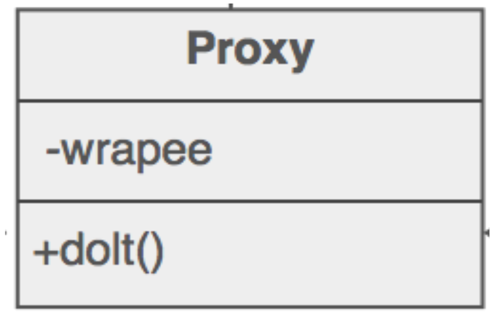


```
//caches frequently requested songs. If a song is not already cached
function SongProxy() {
  var songws = new SongWS();
  return {
    getSong: function(songInput) {
      if (!songcache[songInput]) //cache miss -> add to cache
        songcache[songInput] = songws.getSong(songInput);
      return songcache[songInput];
    },
    getCount: function() {
      var count = 0;
      for (var song in songcache) { count++; }
      return songcache.count;
    }
  };
};
```



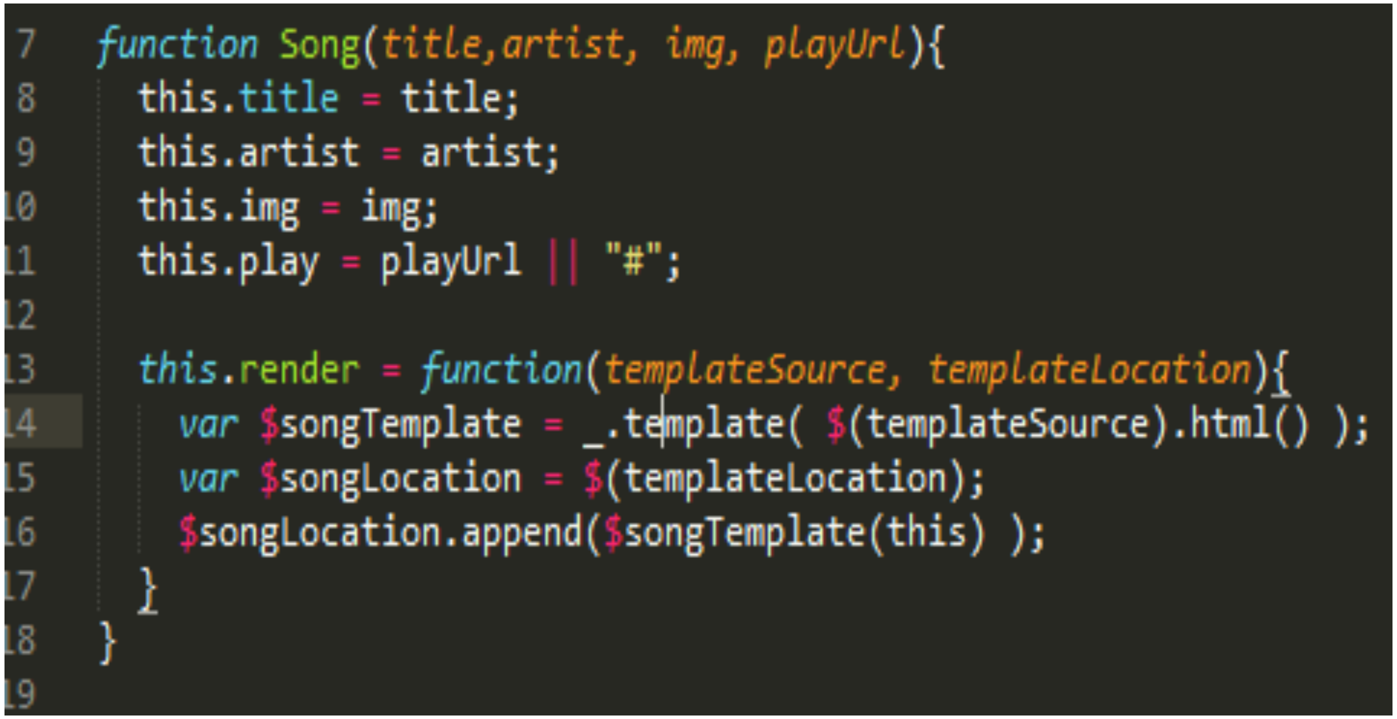
songFinder - Refactoring using Proxy (cont)

- ★ Provides an interface similar to the real object
- ★ Maintains a reference that lets the proxy access the real object
- ★ Handles requests and forwards these to the real object



songFinder - Refactoring using Decorator I

Simple Song Object



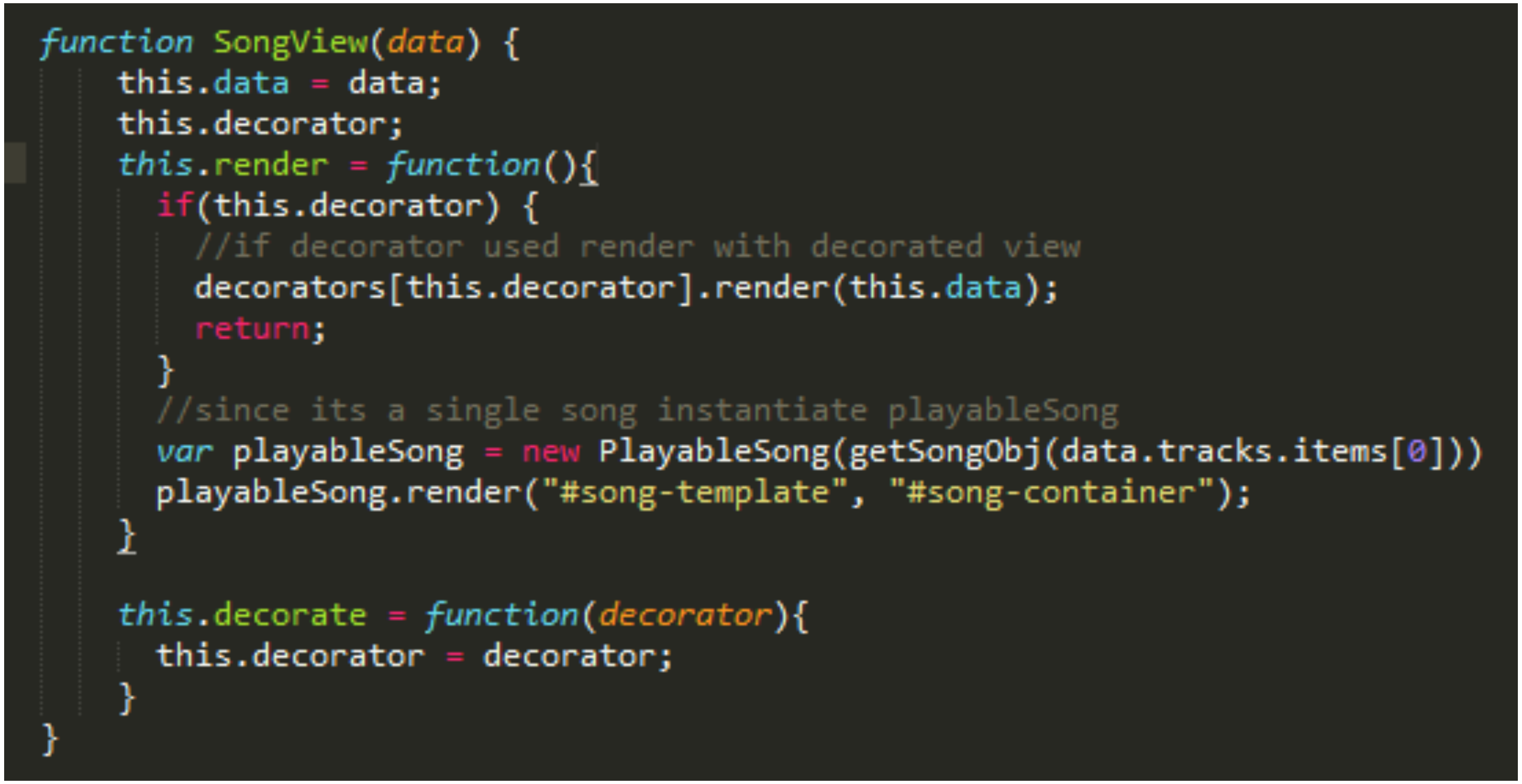
```
function Song(title,artist, img, playUrl){
  this.title = title;
  this.artist = artist;
  this.img = img;
  this.play = playUrl || "#";

  this.render = function(templateSource, templateLocation){
    var $songTemplate = _.template( $(templateSource).html() );
    var $songLocation = $(templateLocation);
    $songLocation.append($songTemplate(this) );
  }
}
```

songFinder - Refactoring using Decorator I

Decorated Song Object - Playable Song

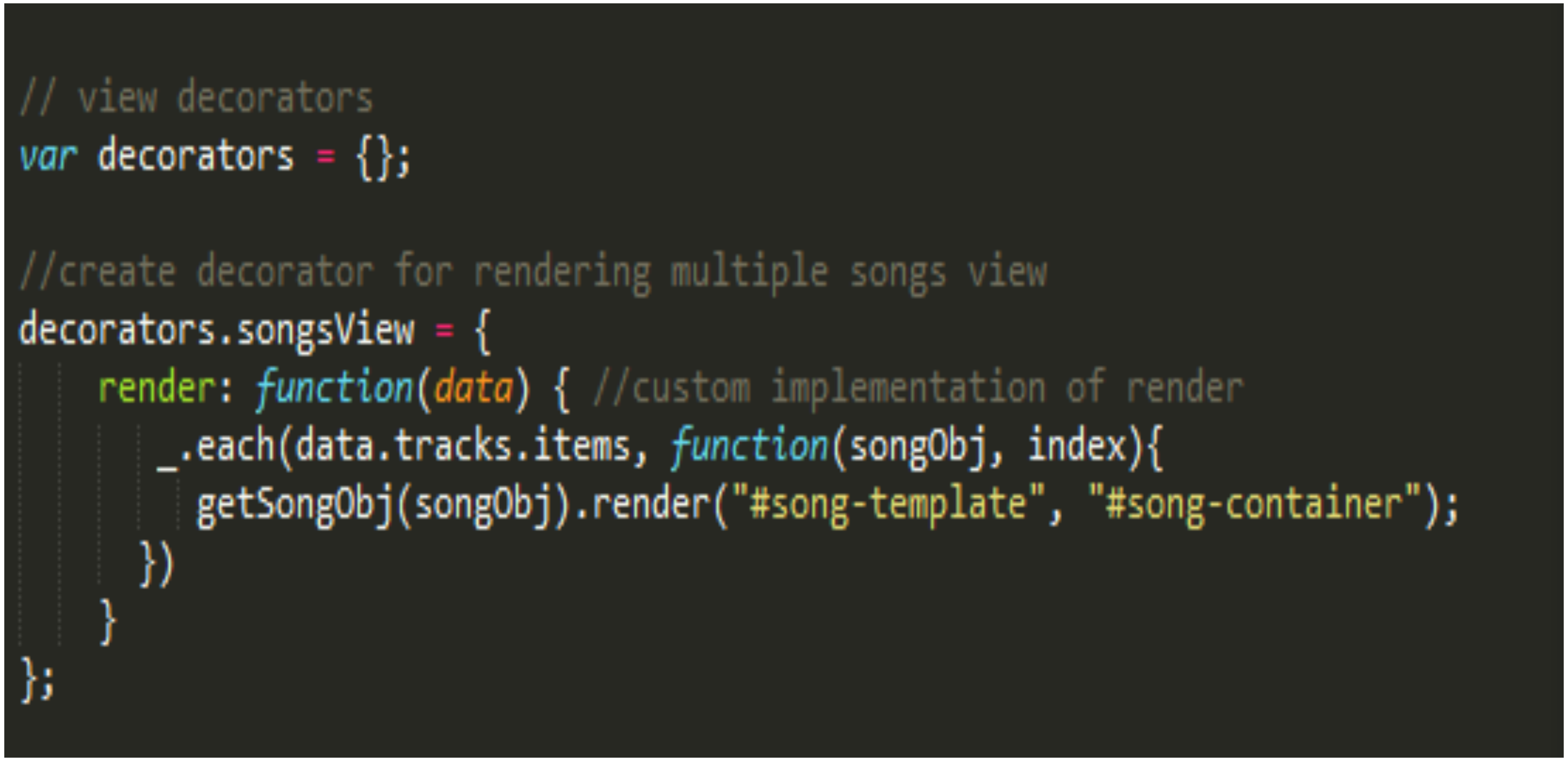
image to be added

songFinder - Refactoring using Decorator II



```
function SongView(data) {
    this.data = data;
    this.decorator;
    this.render = function(){
      if(this.decorator) {
        //if decorator used render with decorated view
        decorators[this.decorator].render(this.data);
        return;
      }
      //since its a single song instantiate playableSong
      var playableSong = new PlayableSong(getSongObj(data.tracks.items[0]))
      playableSong.render("#song-template", "#song-container");
    }

    this.decorate = function(decorator){
      this.decorator = decorator;
    }
}
```

songFinder - Refactoring using Decorator II



```
// view decorators
var decorators = {};

//create decorator for rendering multiple songs view
decorators.songsView = {
    render: function(data) { //custom implementation of render
        _.each(data.tracks.items, function(songObj, index){
            getSongObj(songObj).render("#song-template", "#song-container");
        })
    }
};
```

songFinder - Refactoring using Decorator II

We achieved to:

★ change efficiently the Song object behavior based on the response format at runtime

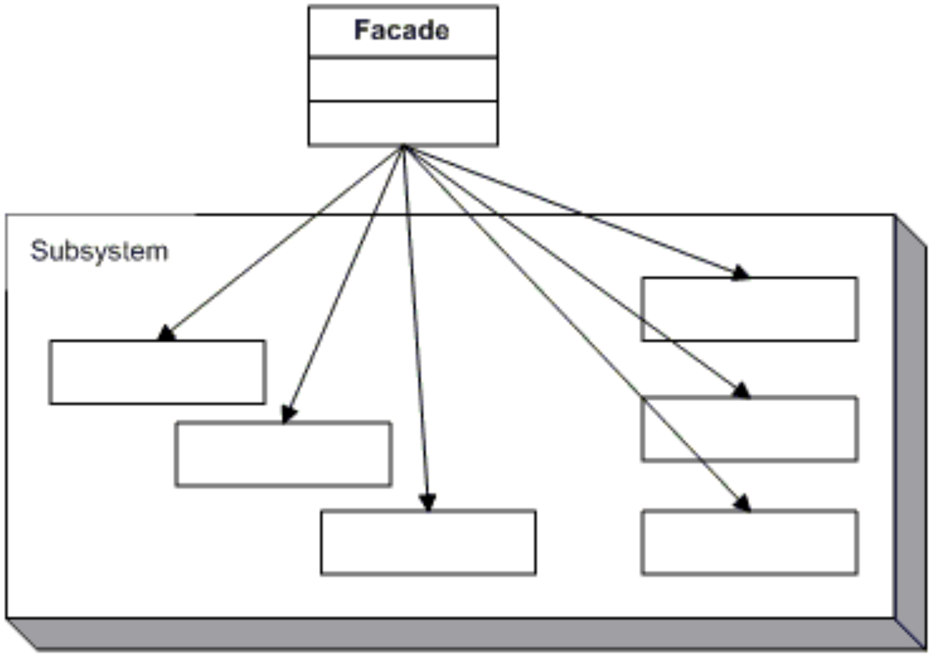
★ provide a clean and object-oriented way to handle and render different server responses

Façade Design Pattern

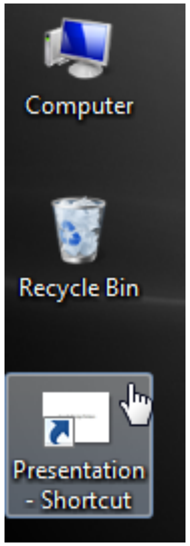
Façade Pattern - Definition

- Convenient higher-level interface to a larger body of code
- Hides actual complexity
- Simplified presentation of API - Improves usability
- Decouples class from code that utilizes it

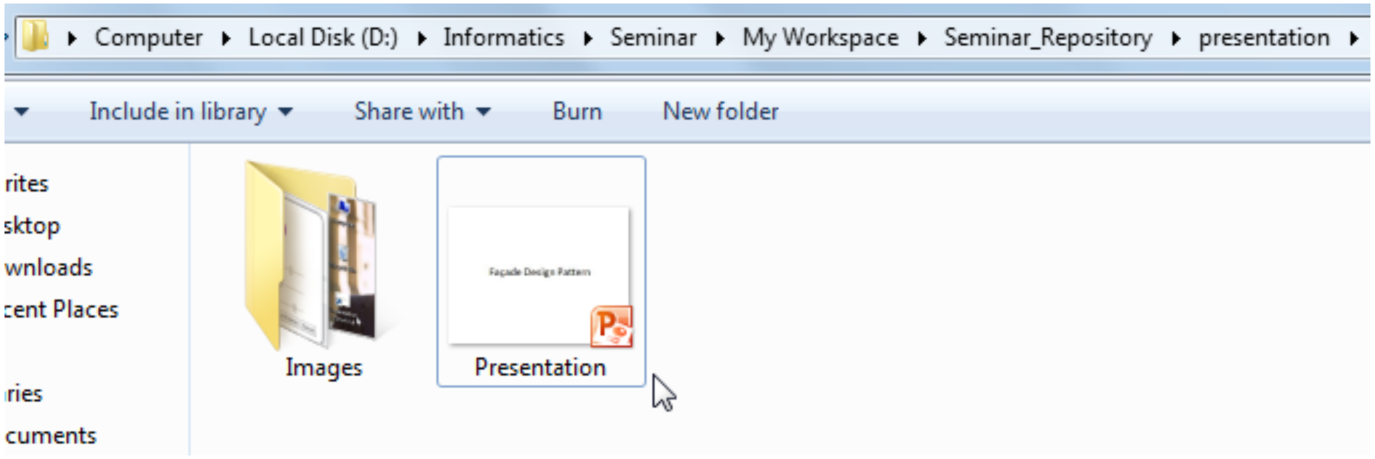
Façade Pattern - Structure



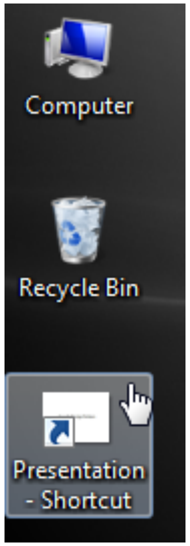
Example



Example



Computer ▸ Local Disk (D:) ▸ Informatics ▸ Seminar ▸ My Workspace ▸ Seminar_Repository ▸ presentation ▸
Include in library
Share with
Burn
New folder
rites
sktop
wnloads
cent Places
ries
cuments
Images
Presentation



Computer
Recycle Bin
Presentation - Shortcut

Example - jQuery

Group of facades – makes programming easier and faster

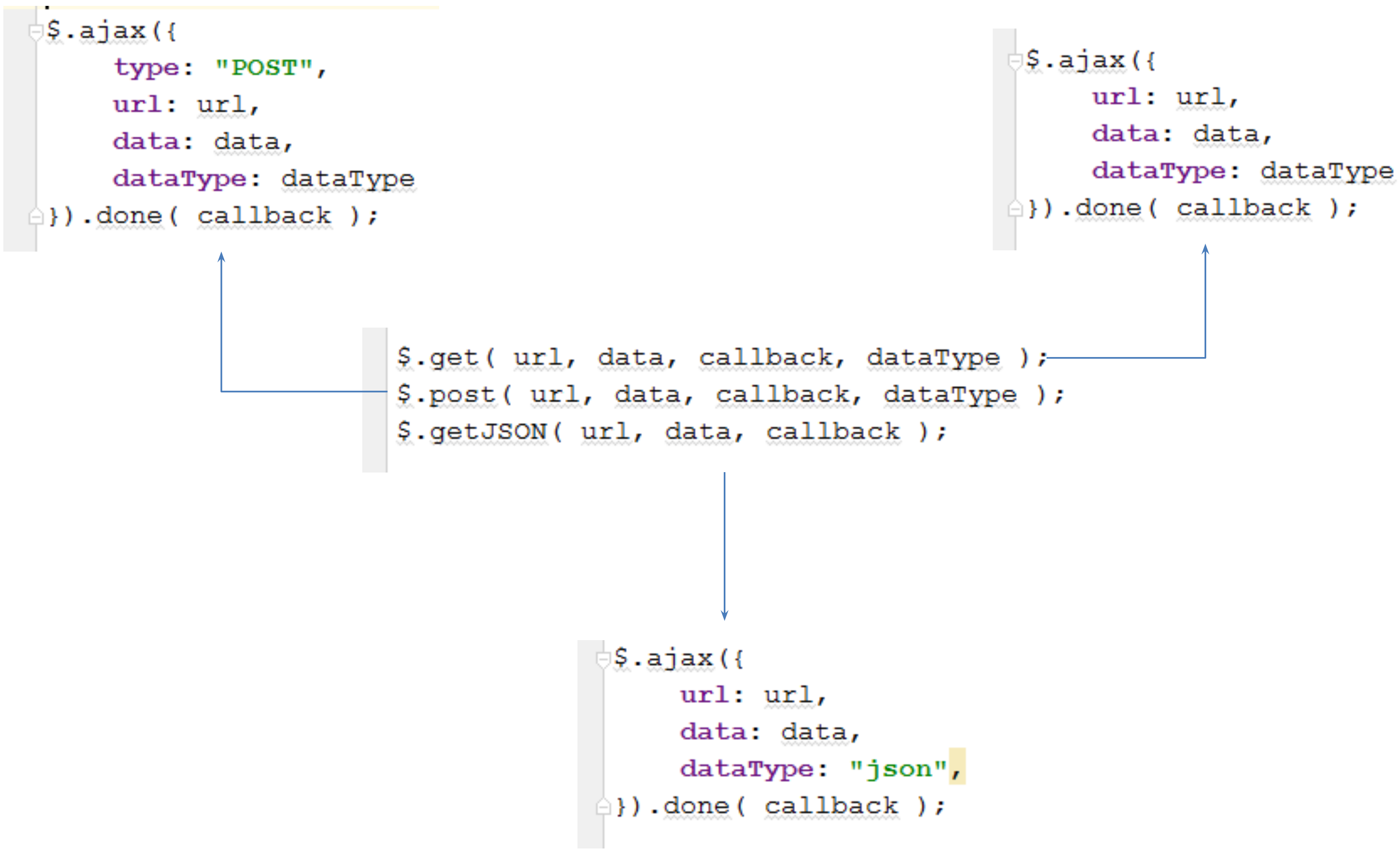
Example - jQuery

Group of facades – makes programming easier and faster

facades for $.ajax()

- $.get(url, data, callback, dataType);
- $.post(url, data, callback, dataType);
- $.getJSON(url, data, callback);

Behind the scenes



```
$.ajax({
    type: "POST",
    url: url,
    data: data,
    dataType: dataType
}).done( callback );
```

```
$.ajax({
    url: url,
    data: data,
    dataType: dataType
}).done( callback );
```

```
$.get( url, data, callback, dataType );
$.post( url, data, callback, dataType );
$.getJSON( url, data, callback );
```

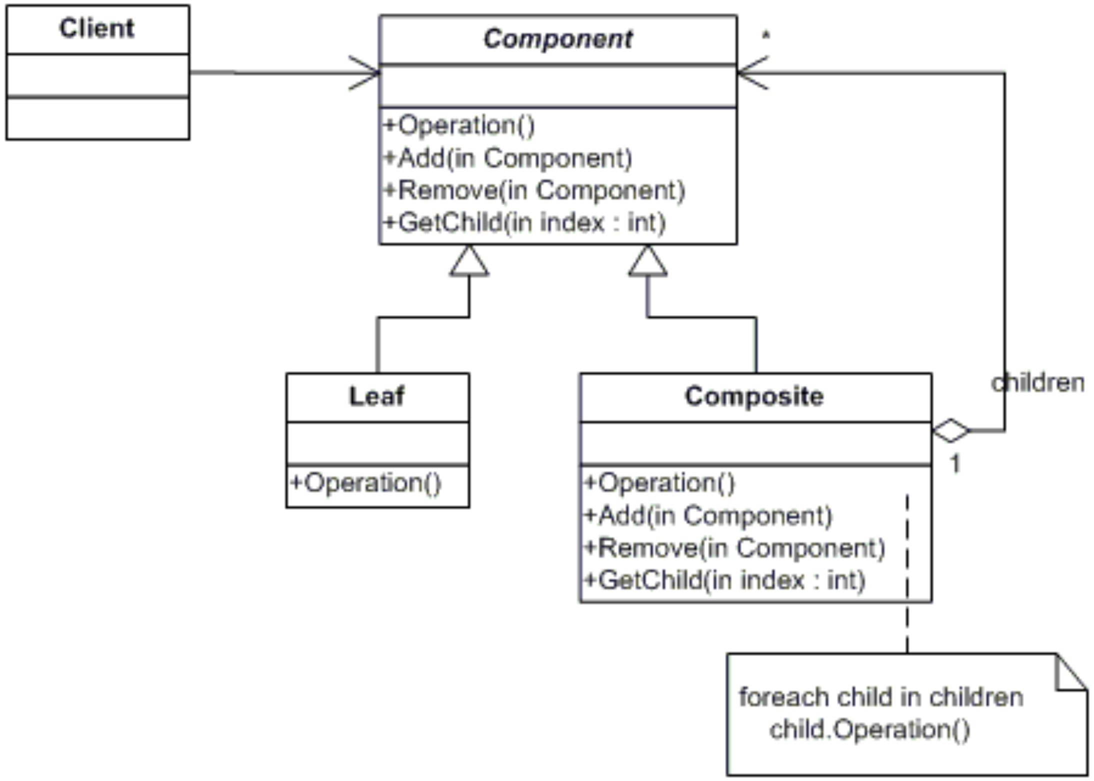
```
$.ajax({
    url: url,
    data: data,
    dataType: "json",
}).done( callback );
```

Composite Design Pattern

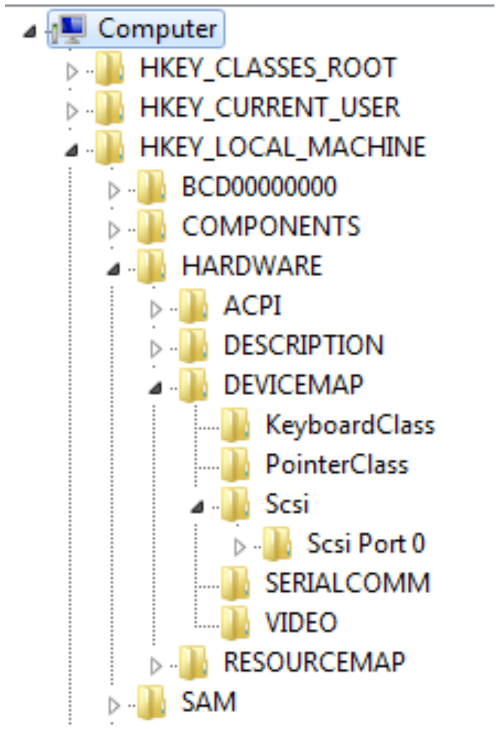
Composite Pattern - Introduction

- Group of objects should be treated in the same way as a single instance of an object.
- Same behavior applied to an object or a group of objects
- Code resuability

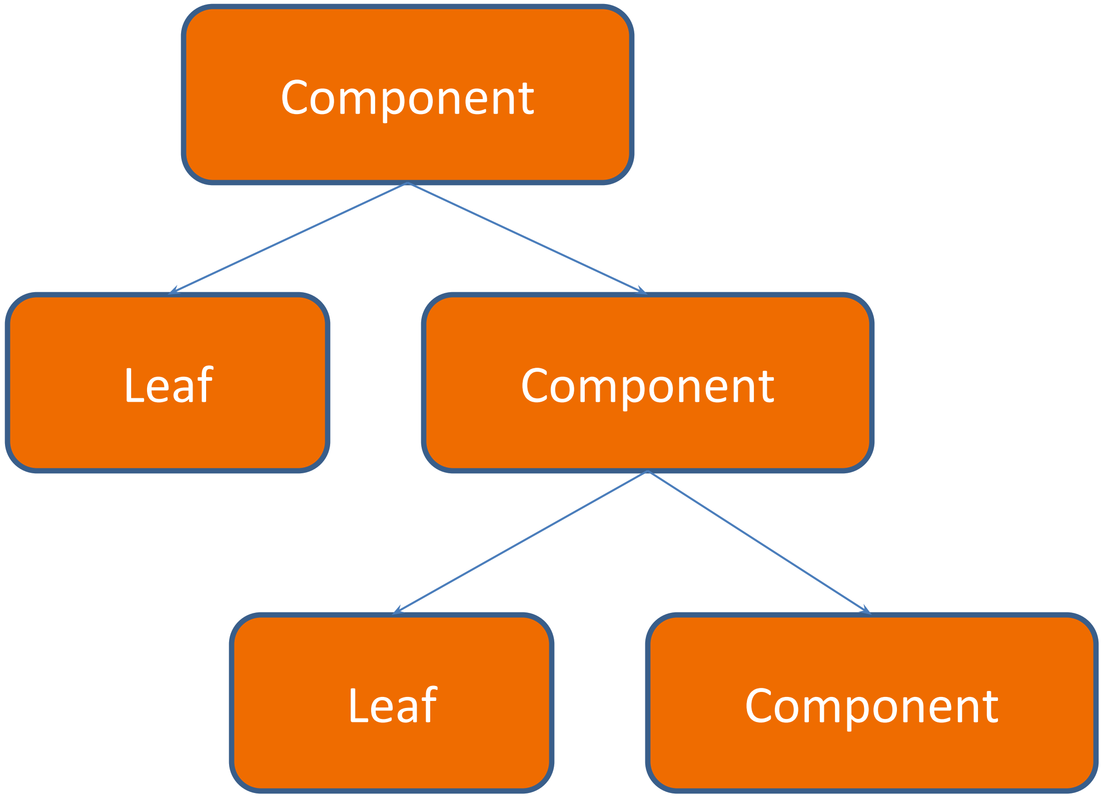
Composite Pattern - Structure



Example



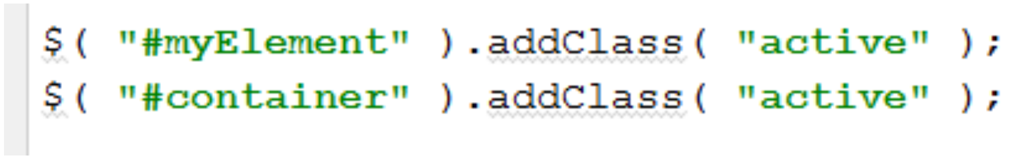
```
Computer
  HKEY_CLASSES_ROOT
  HKEY_CURRENT_USER
  HKEY_LOCAL_MACHINE
    BCD00000000
    COMPONENTS
    HARDWARE
      ACPI
      DESCRIPTION
      DEVICEMAP
        KeyboardClass
        PointerClass
        Scsi
          Scsi Port 0
        SERIALCOMM
        VIDEO
      RESOURCEMAP
    SAM
```



Example - jQuery

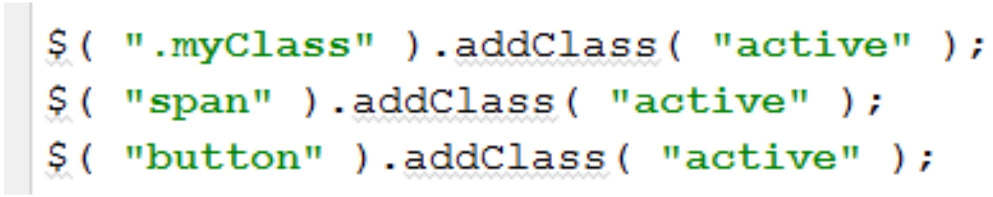
Same methods available to be applied to an element or collection of elements

Single Element



```
$( "#myElement" ).addClass( "active" );
$( "#container" ).addClass( "active" );
```

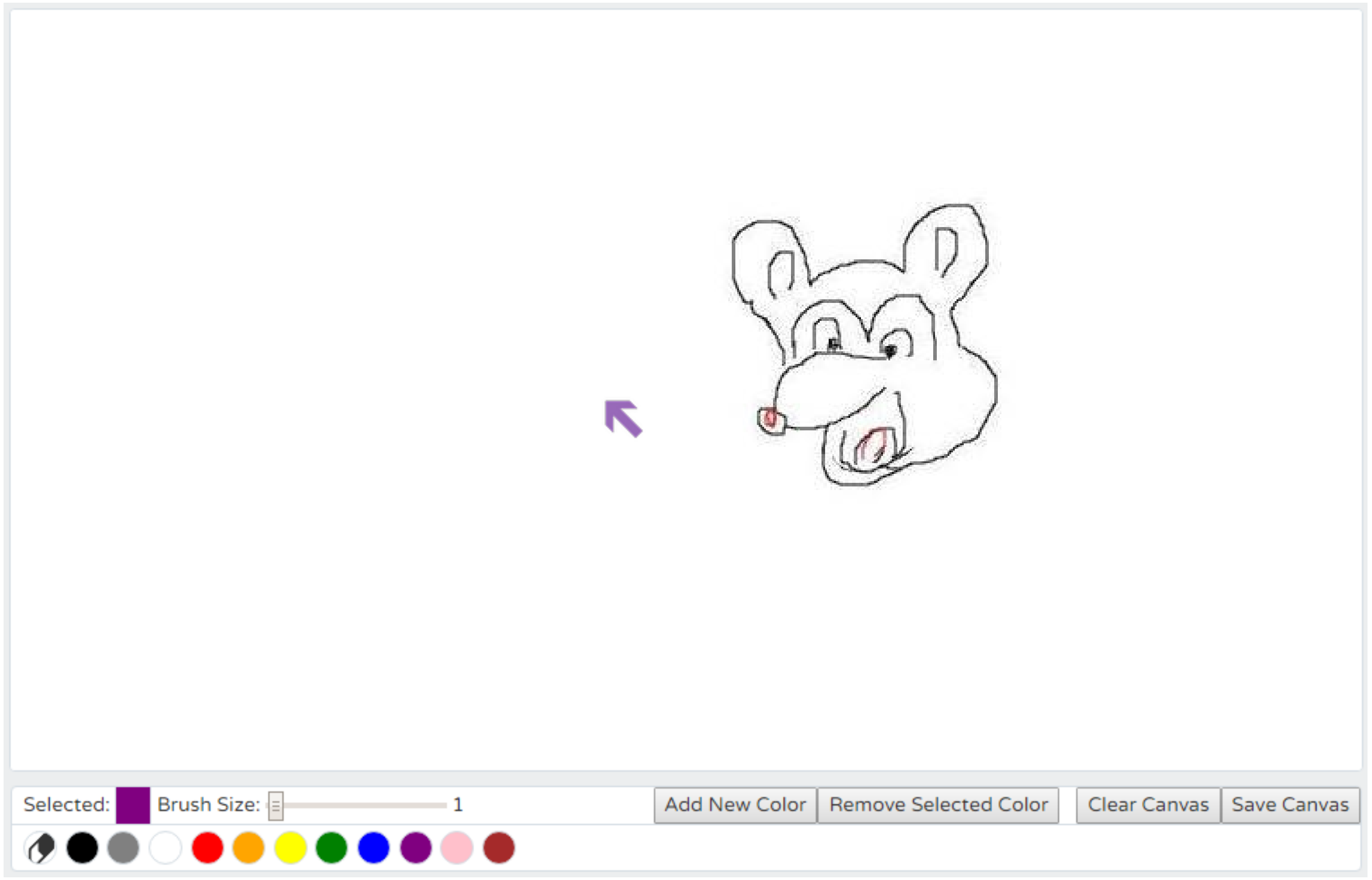
Collection of Elements



```
$( ".myClass" ).addClass( "active" );
$( "span" ).addClass( "active" );
$( "button" ).addClass( "active" );
```

Project : drawr-bootstrap[1]

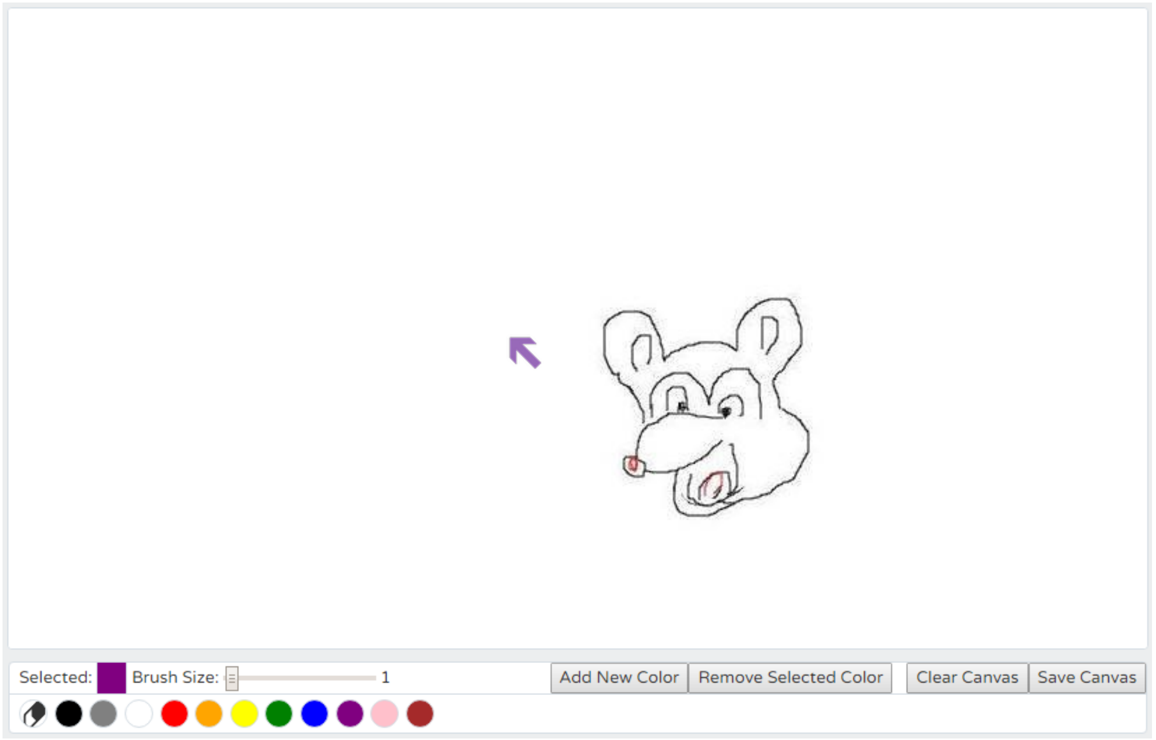
drawr-bootstrap



First Screen

drawr-bootstrap: features

- Customizable drawing tool
- Toolbar
  - Brush/Eraser thickness
  - Color Palette
  - Add new color
  - Remove a color
  - Clear canvas
  - Save canvas

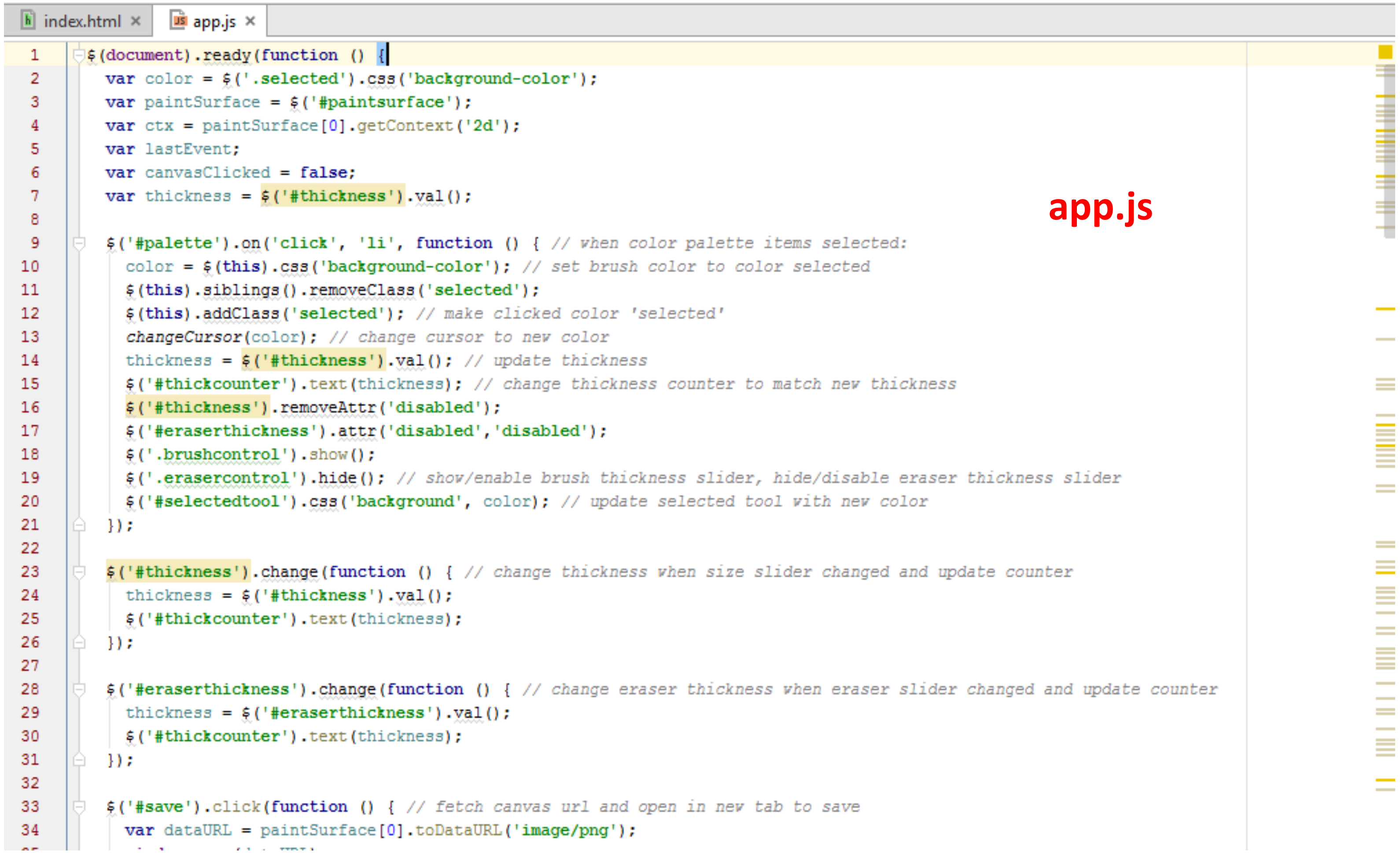


Applying Façade Design Pattern

Several event listeners & handlers

- $('**#palette**').on('**click**', '**li**', **function** () {…}
- $('**#erase**').click(**function** () { …}
- $('**#thickness**').change(**function** () {…}
- $('**#eraserthickness**').change(**function** () {…}
- $('**#save**').click(**function** () {…}
- $('**#clear**').click(**function** () {…}
- $('**#addcolor**').click(**function** () {…}
- $('**#removecolor**').click(**function** () {…}
- $('**#attachcolor**').click(**function** () {…}
- $('**#cancelcolor**').click(**function** () {…}
- $('**.colorslider**').change(**function** () {…}

Original Version



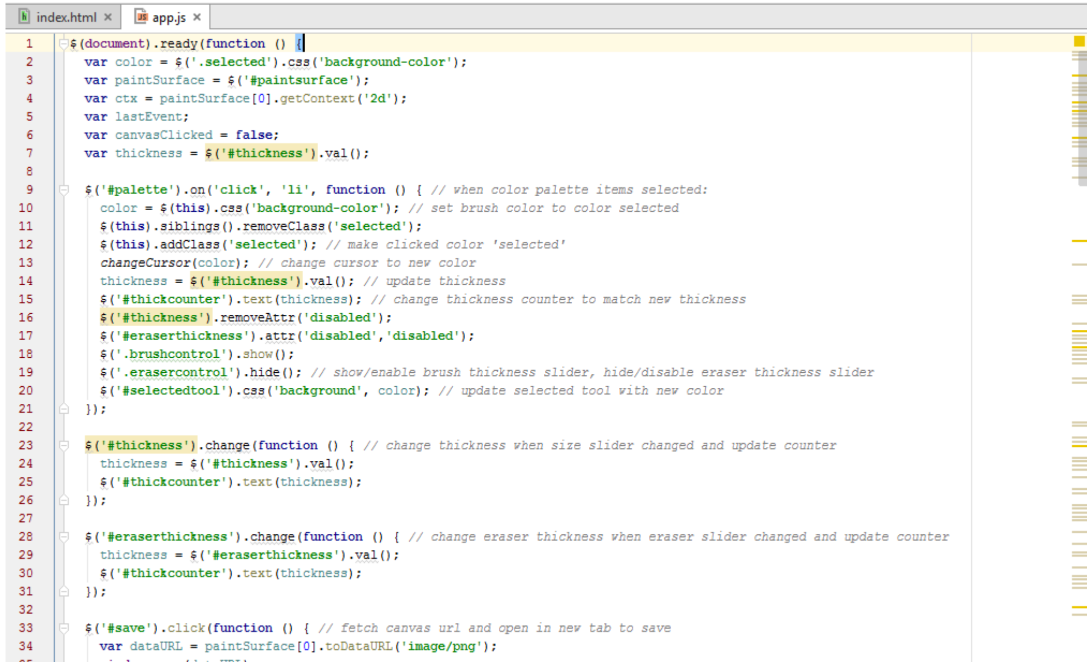
```
$(document).ready(function () {
  var color = $('.selected').css('background-color');
  var paintSurface = $('#paintsurface');
  var ctx = paintSurface[0].getContext('2d');
  var lastEvent;
  var canvasClicked = false;
  var thickness = $('#thickness').val();

  $('#palette').on('click', 'li', function () { // when color palette items selected:
    color = $(this).css('background-color'); // set brush color to color selected
    $(this).siblings().removeClass('selected');
    $(this).addClass('selected'); // make clicked color 'selected'
    changeCursor(color); // change cursor to new color
    thickness = $('#thickness').val(); // update thickness
    $('#thickcounter').text(thickness); // change thickness counter to match new thickness
    $('#thickness').removeAttr('disabled');
    $('#eraserthickness').attr('disabled','disabled');
    $('.brushcontrol').show();
    $('.erasercontrol').hide(); // show/enable brush thickness slider, hide/disable eraser thickness slider
    $('#selectedtool').css('background', color); // update selected tool with new color
  });

  $('#thickness').change(function () { // change thickness when size slider changed and update counter
    thickness = $('#thickness').val();
    $('#thickcounter').text(thickness);
  });

  $('#eraserthickness').change(function () { // change eraser thickness when eraser slider changed and update counter
    thickness = $('#eraserthickness').val();
    $('#thickcounter').text(thickness);
  });

  $('#save').click(function () { // fetch canvas url and open in new tab to save
    var dataURL = paintSurface[0].toDataURL('image/png');
```

app.js

Limitations

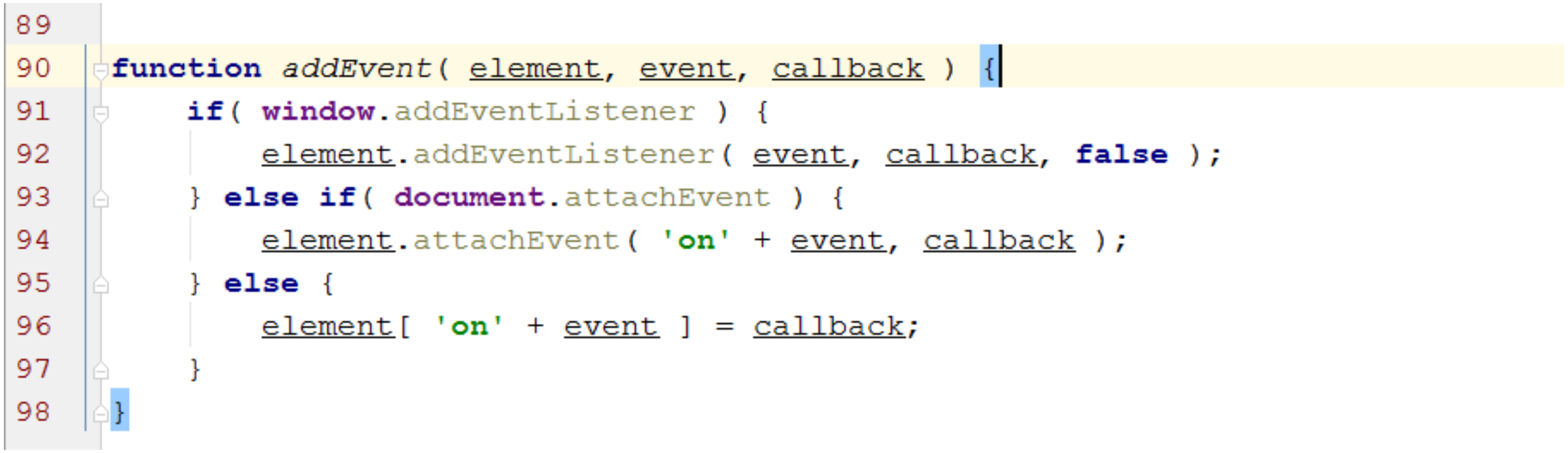
- All event listeners/handlers defined in a single function
- Confusing! No abstraction
- Browsers’ compatibility not checked

app.js



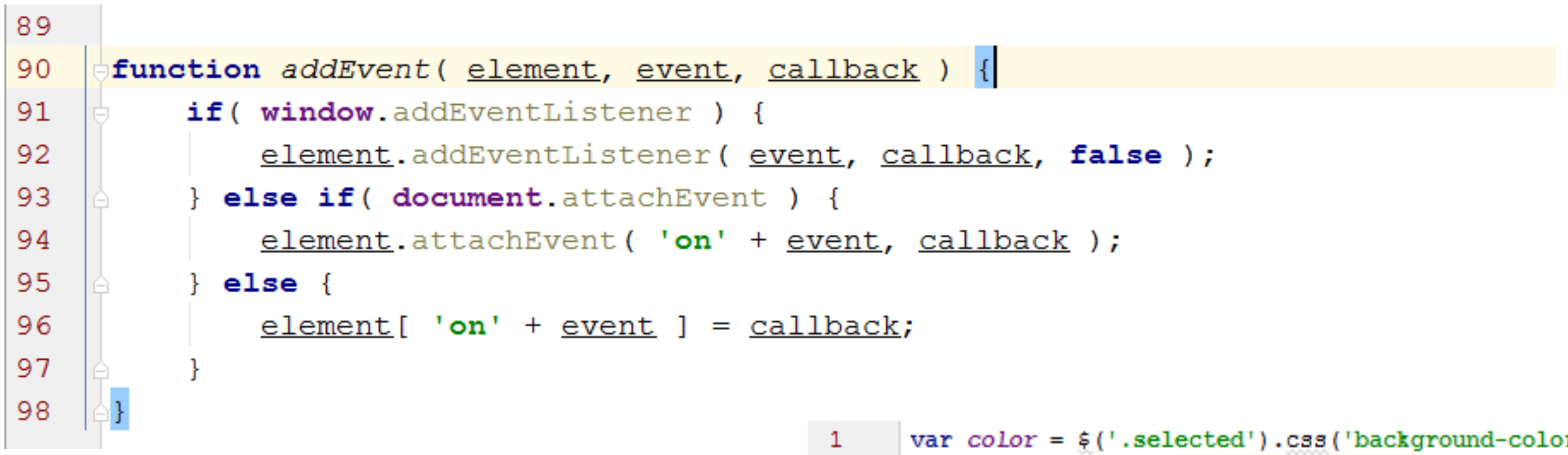
```
$(document).ready(function () {
  var color = $('.selected').css('background-color');
  var paintSurface = $('#paintsurface');
  var ctx = paintSurface[0].getContext('2d');
  var lastEvent;
  var canvasClicked = false;
  var thickness = $('#thickness').val();

  $('#palette').on('click', 'li', function () { // when color palette items selected:
    color = $(this).css('background-color'); // set brush color to color selected
    $(this).siblings().removeClass('selected');
    $(this).addClass('selected'); // make clicked color 'selected'
    changeCursor(color); // change cursor to new color
    thickness = $('#thickness').val(); // update thickness
    $('#thickcounter').text(thickness); // change thickness counter to match new thickness
    $('#thickness').removeAttr('disabled');
    $('#eraserthickness').attr('disabled','disabled');
    $('.brushcontrol').show();
    $('.erasercontrol').hide(); // show/enable brush thickness slider, hide/disable eraser thickness slider
    $('#selectedtool').css('background', color); // update selected tool with new color
  });

  $('#thickness').change(function () { // change thickness when size slider changed and update counter
    thickness = $('#thickness').val();
    $('#thickcounter').text(thickness);
  });

  $('#eraserthickness').change(function () { // change eraser thickness when eraser slider changed and update counter
    thickness = $('#eraserthickness').val();
    $('#thickcounter').text(thickness);
  });

  $('#save').click(function () { // fetch canvas url and open in new tab to save
    var dataURL = paintSurface[0].toDataURL('image/png');
```

a simple facade that masks the various browser-specific methods



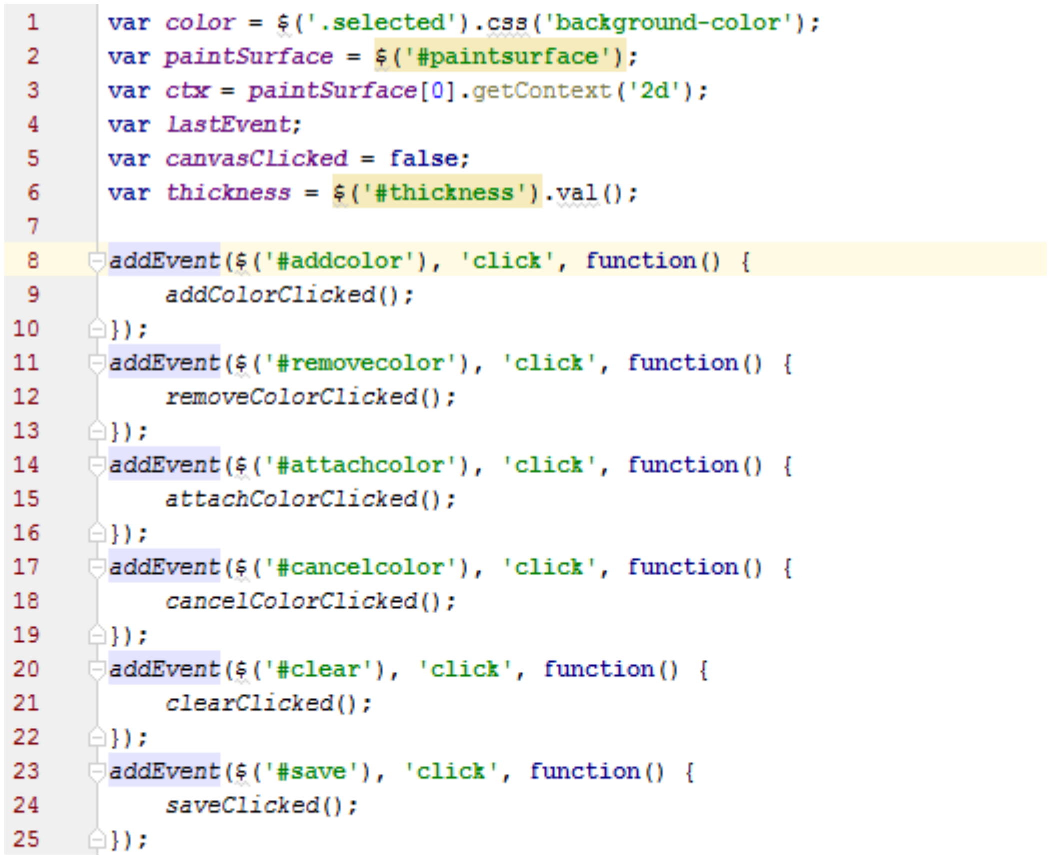
```
function addEvent( element, event, callback ) {
    if( window.addEventListener ) {
        element.addEventListener( event, callback, false );
    } else if( document.attachEvent ) {
        element.attachEvent( 'on' + event, callback );
    } else {
        element[ 'on' + event ] = callback;
    }
}
```

app.js



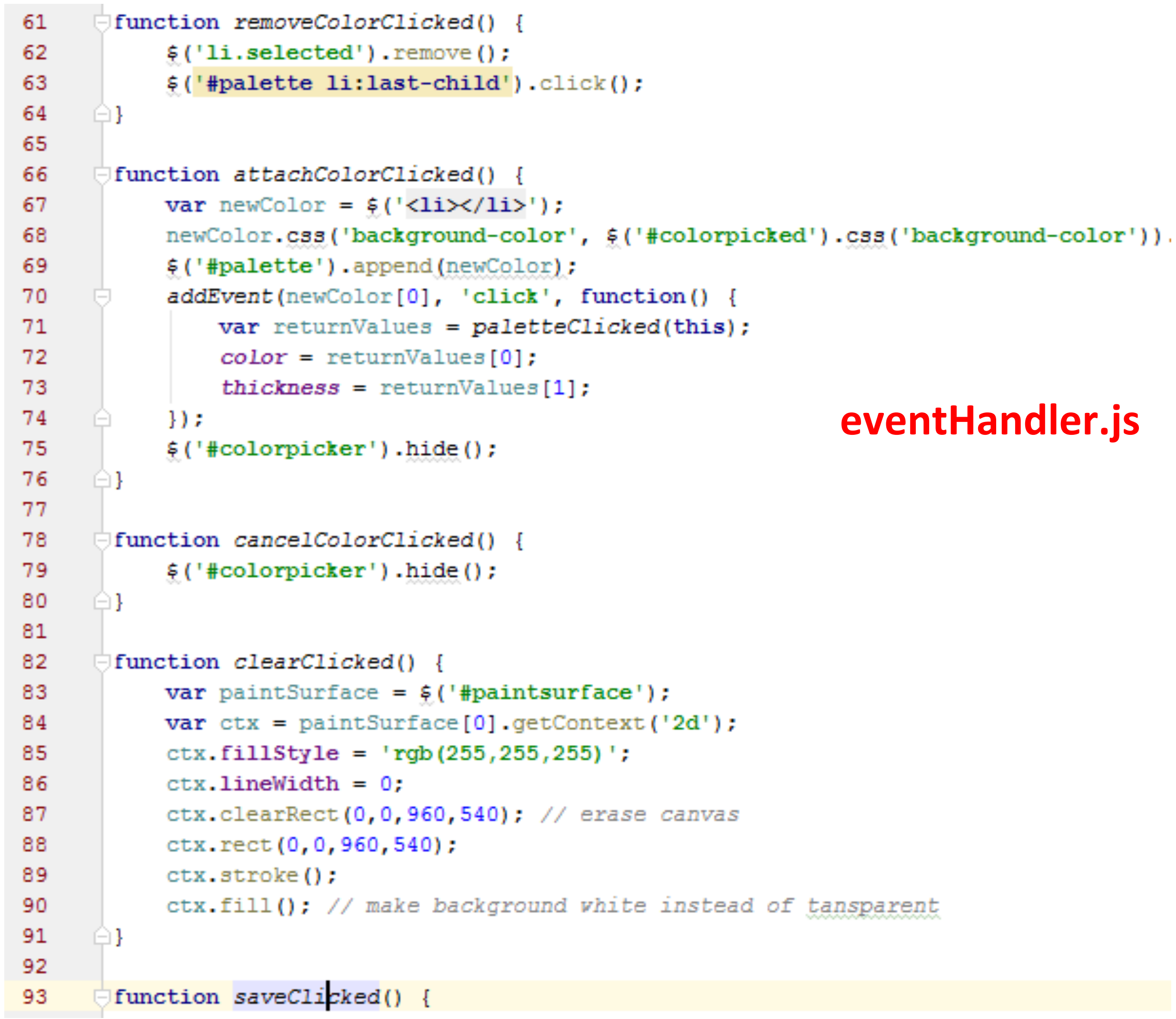
```
function addEvent( element, event, callback ) {
    if( window.addEventListener ) {
        element.addEventListener( event, callback, false );
    } else if( document.attachEvent ) {
        element.attachEvent( 'on' + event, callback );
    } else {
        element[ 'on' + event ] = callback;
    }
}
```

app.js



```
var color = $('.selected').css('background-color');
var paintSurface = $('#paintsurface');
var ctx = paintSurface[0].getContext('2d');
var lastEvent;
var canvasClicked = false;
var thickness = $('#thickness').val();

addEvent($('#addcolor'), 'click', function() {
    addColorClicked();
});
addEvent($('#removecolor'), 'click', function() {
    removeColorClicked();
});
addEvent($('#attachcolor'), 'click', function() {
    attachColorClicked();
});
addEvent($('#cancelcolor'), 'click', function() {
    cancelColorClicked();
});
addEvent($('#clear'), 'click', function() {
    clearClicked();
});
addEvent($('#save'), 'click', function() {
    saveClicked();
});
```

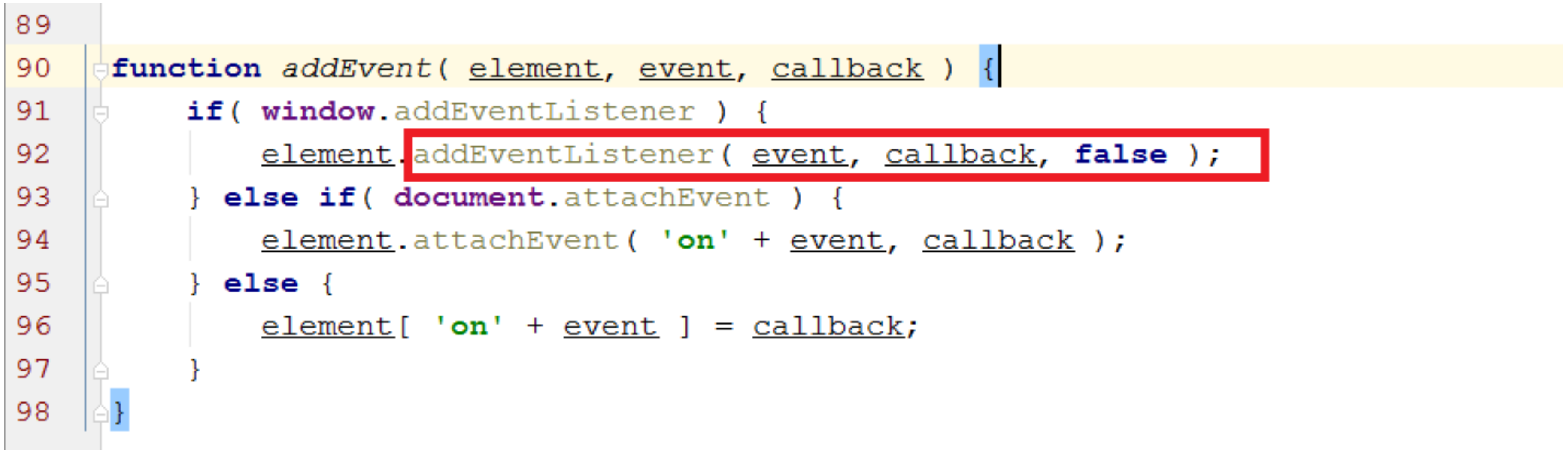
Achieving abstraction



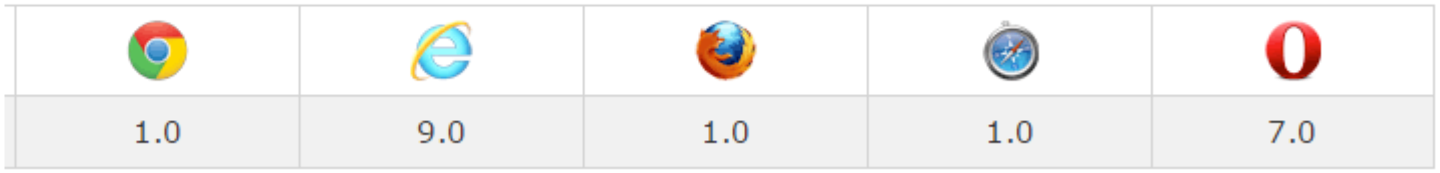
```
function removeColorClicked() {
    $('li.selected').remove();
    $('#palette li:last-child').click();
}

function attachColorClicked() {
    var newColor = $('<li></li>');
    newColor.css('background-color', $('#colorpicked').css('background-color')).
    $('#palette').append(newColor);
    addEvent(newColor[0], 'click', function() {
        var returnValues = paletteClicked(this);
        color = returnValues[0];
        thickness = returnValues[1];
    });
    $('#colorpicker').hide();
}

function cancelColorClicked() {
    $('#colorpicker').hide();
}

function clearClicked() {
    var paintSurface = $('#paintsurface');
    var ctx = paintSurface[0].getContext('2d');
    ctx.fillStyle = 'rgb(255,255,255)';
    ctx.lineWidth = 0;
    ctx.clearRect(0,0,960,540); // erase canvas
    ctx.rect(0,0,960,540);
    ctx.stroke();
    ctx.fill(); // make background white instead of tansparent
}

function saveClicked() {
```

eventHandler.js

Browser-specific methods



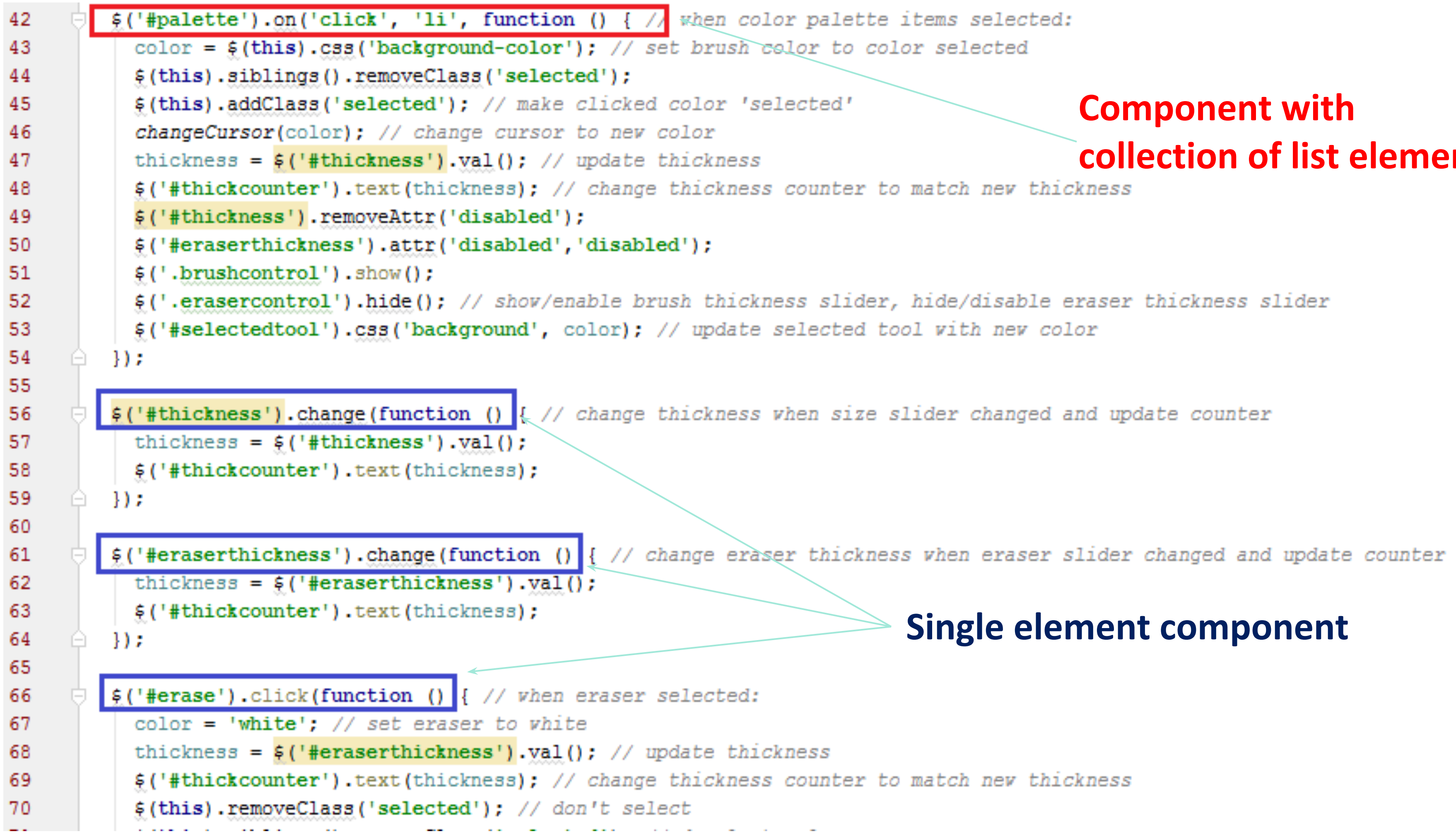
```
function addEvent( element, event, callback ) {
    if( window.addEventListener ) {
        element.addEventListener( event, callback, false );
    } else if( document.attachEvent ) {
        element.attachEvent( 'on' + event, callback );
    } else {
        element[ 'on' + event ] = callback;
    }
}
```



1.0	9.0	1.0	1.0	7.0

Applying Composite Design Pattern

Original version

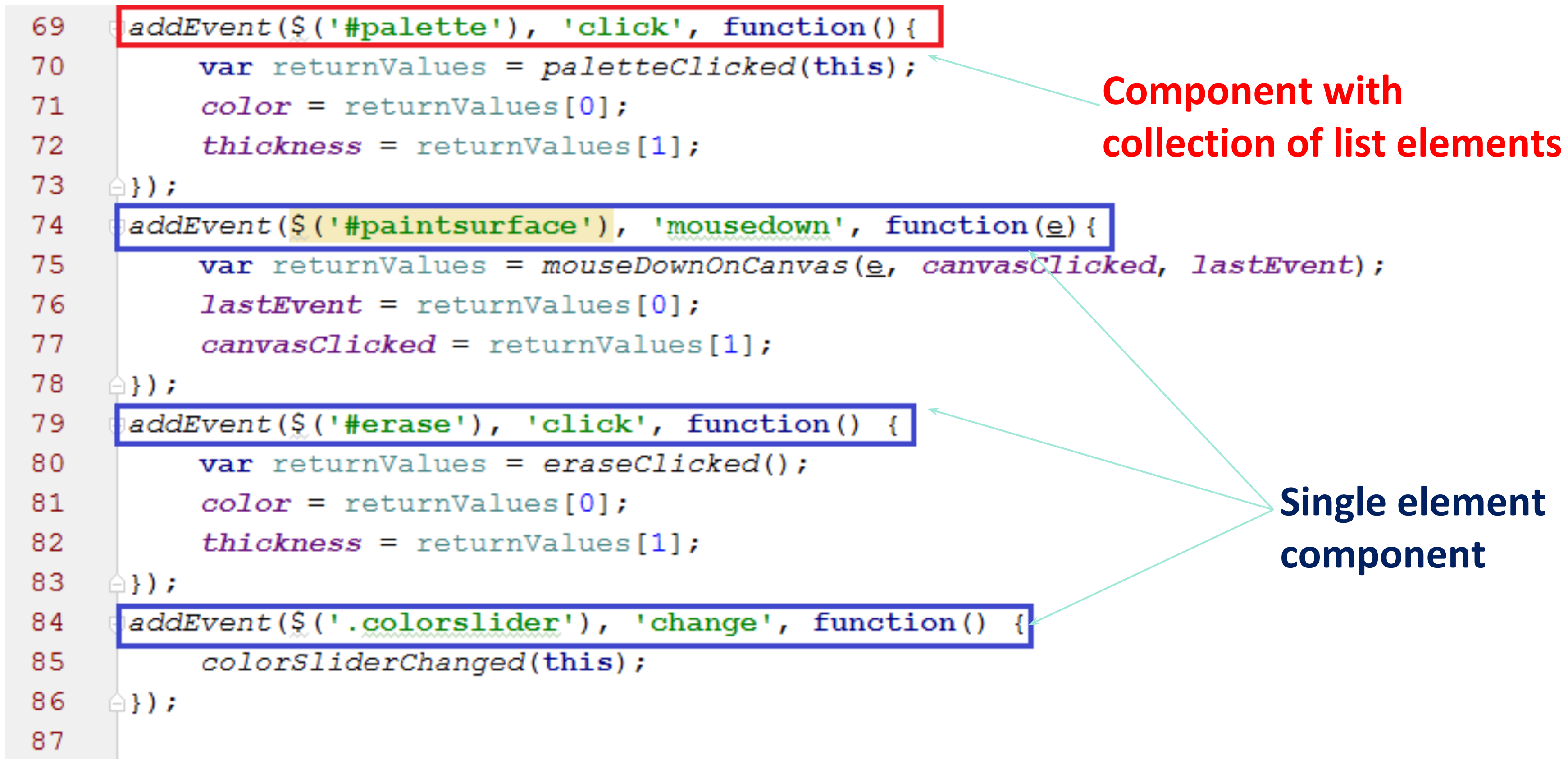


```
$('#palette').on('click', 'li', function () { // when color palette items selected:
    color = $(this).css('background-color'); // set brush color to color selected
    $(this).siblings().removeClass('selected');
    $(this).addClass('selected'); // make clicked color 'selected'
    changeCursor(color); // change cursor to new color
    thickness = $('#thickness').val(); // update thickness
    $('#thickcounter').text(thickness); // change thickness counter to match new thickness
    $('#thickness').removeAttr('disabled');
    $('#eraserthickness').attr('disabled','disabled');
    $('.brushcontrol').show();
    $('.erasercontrol').hide(); // show/enable brush thickness slider, hide/disable eraser thickness slider
    $('#selectedtool').css('background', color); // update selected tool with new color
});

$('#thickness').change(function () { // change thickness when size slider changed and update counter
    thickness = $('#thickness').val();
    $('#thickcounter').text(thickness);
});

$('#eraserthickness').change(function () { // change eraser thickness when eraser slider changed and update counter
    thickness = $('#eraserthickness').val();
    $('#thickcounter').text(thickness);
});

$('#erase').click(function () { // when eraser selected:
    color = 'white'; // set eraser to white
    thickness = $('#eraserthickness').val(); // update thickness
    $('#thickcounter').text(thickness); // change thickness counter to match new thickness
    $(this).removeClass('selected'); // don't select
```

Component with collection of list elements

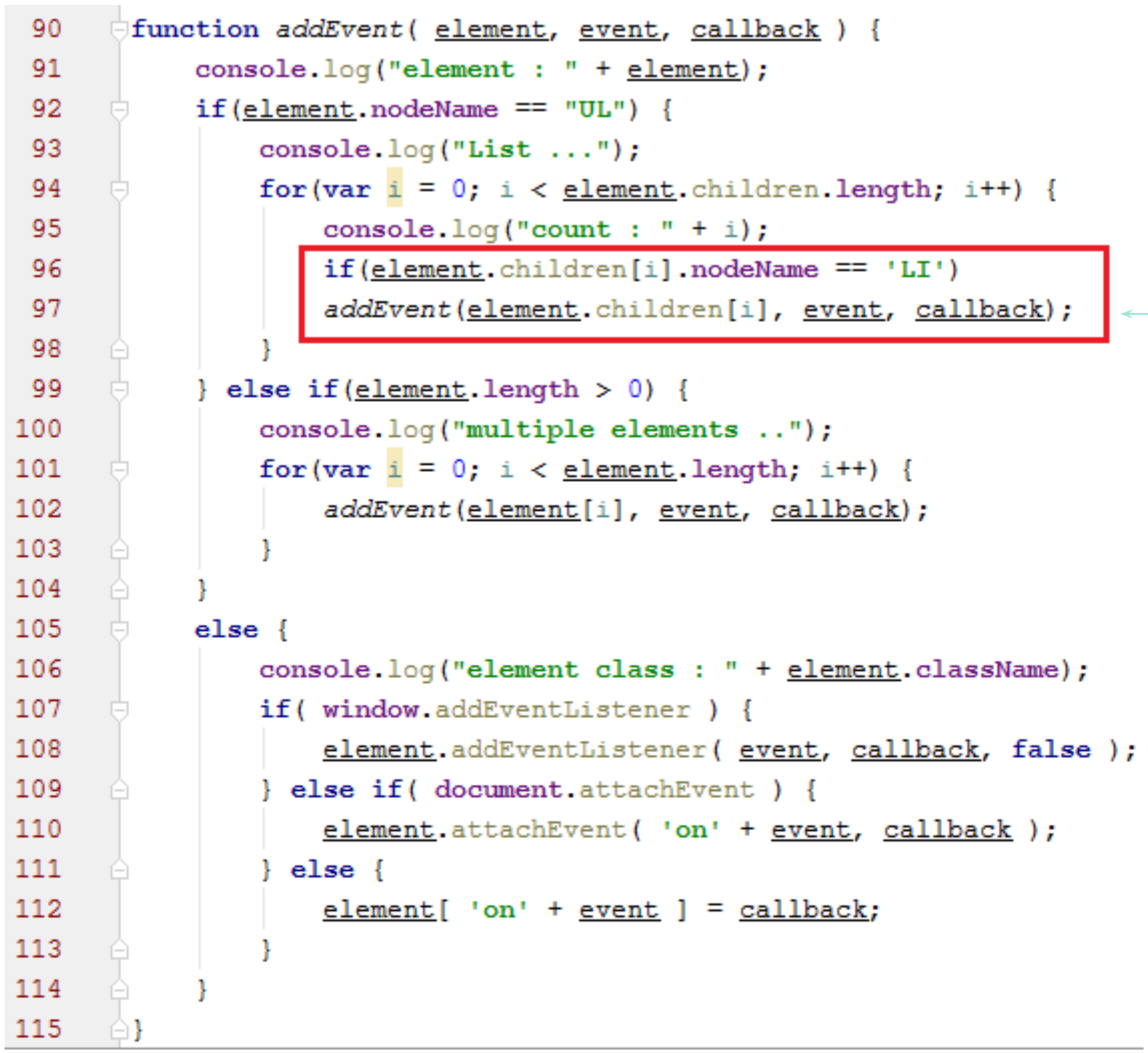
Single element component

Refactored version



```
addEvent($('#palette'), 'click', function(){
        var returnValues = paletteClicked(this);
        color = returnValues[0];
        thickness = returnValues[1];
});
addEvent($('#paintsurface'), 'mousedown', function(e){
        var returnValues = mouseDownOnCanvas(e, canvasClicked, lastEvent);
        lastEvent = returnValues[0];
        canvasClicked = returnValues[1];
});
addEvent($('#erase'), 'click', function() {
        var returnValues = eraseClicked();
        color = returnValues[0];
        thickness = returnValues[1];
});
addEvent($('.colorslider'), 'change', function() {
        colorSliderChanged(this);
});
```

Refactored version



```
function addEvent( element, event, callback ) {
    console.log("element : " + element);
    if(element.nodeName == "UL") {
        console.log("List ...");
        for(var i = 0; i < element.children.length; i++) {
            console.log("count : " + i);
            if(element.children[i].nodeName == 'LI')
            addEvent(element.children[i], event, callback);
        }
    } else if(element.length > 0) {
        console.log("multiple elements ..");
        for(var i = 0; i < element.length; i++) {
            addEvent(element[i], event, callback);
        }
    }
    else {
        console.log("element class : " + element.className);
        if( window.addEventListener ) {
            element.addEventListener( event, callback, false );
        } else if( document.attachEvent ) {
            element.attachEvent( 'on' + event, callback );
        } else {
            element[ 'on' + event ] = callback;
        }
    }
}
```

Recursion for Collection components

References

[1] drawr-bootstrap

https://github.com/flamingveggies/drawr-bootstrap

[2] Sourcemaking

https://sourcemaking.com/design_patterns/structural_patterns

[3] songFinder

https://github.com/goodbedford/songFinder

[4] dofactory

http://www.dofactory.com/javascript/design-patterns

[5] gofpatterns

http://www.gofpatterns.com

Q&A

Thank you

Questions?

Chrysa Papadaki // chr.papadaki@tum.de

Nishant Gupta // nishant.gupta@tum.de