Technical University of Moldova

Chair Computer Science

Report

On Designing Informational Systems

Laboratory Work Nr. 3

Done by: st. gr. FAF-091 Melniciuc Constanin
Bibilici Victor

Checked by: Zarea Ivan

Chişinău – 2012

1. Code Inspection

Code inspection is a largly used technique that have to be used by every coder on this planet. You have to check your code for other people to understand your code, although when you watch someone's code you see fig1.; anyway you have to use some code inspection tools.

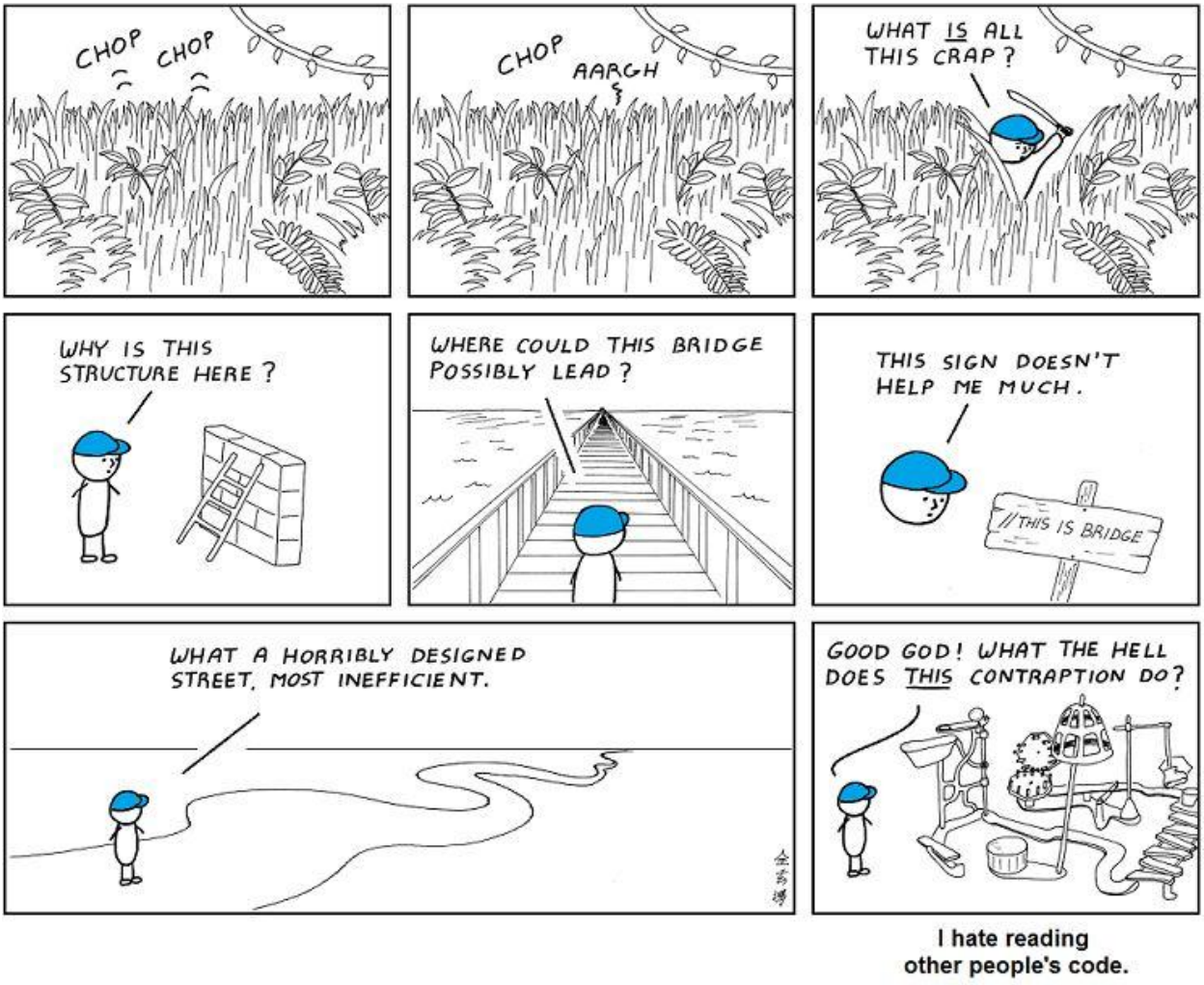

Fig.1-Reading other people's code;

In my application I used Closure Compiler, unfortunately this is a static code analysis but still I got something.

Closure Compiler Theory

The Closure Compiler is a tool for making JavaScript download and run faster. It is a true compiler for JavaScript. Instead of compiling from a source language to machine code, it compiles from JavaScript to better JavaScript. It parses your JavaScript, analyzes it,

removes dead code and rewrites and minimizes what's left. It also checks syntax, variable references, and types, and warns about common JavaScript pitfalls.

The Closure Tools project is an effort by Google engineers to open source the tools used in many of Google's sites and web applications for use by the wider Web development community.

Web applications have evolved from simple HTML pages into rich, interactive applications that provide a great user experience.

Initial Code

```
(function( $ ){
	$.fn.qrcode = function(options) {
		if( typeof options === 'string' ){
			options		= { text: options };
		}
		options		= $.extend( {}, {
			render		: "canvas",
			width		: 256,
			height		: 256,
			typeNumber	: -1,
			correctLevel	: QRErrorCorrectLevel.H,
		background	: "#ffffff",
		foreground	: "#000000"
		}, options);

		var createCanvas	= function(){
			var qrcode	= new QRCode(options.typeNumber,
options.correctLevel);
			qrcode.addData(options.text);
			qrcode.make();
			var canvas	= document.createElement('canvas');
			canvas.width	= options.width;
			canvas.height	= options.height;
			var ctx		= canvas.getContext('2d');
			var tileW	= options.width  / qrcode.getModuleCount();
			var tileH	= options.height / qrcode.getModuleCount();
			for( var row = 0; row < qrcode.getModuleCount(); row++ ){
				for( var col = 0; col < qrcode.getModuleCount(); col++ ){
					ctx.fillStyle = qrcode.isDark(row, col) ?
options.foreground : options.background;
					var w = (Math.ceil((col+1)*tileW) -
Math.floor(col*tileW));
					var h = (Math.ceil((row+1)*tileW) -
Math.floor(row*tileW));
```

```
        ctx.fillRect(Math.round(col*tileW),Math.round(row*tileH), w, h);
                                }
                        }
                        return canvas;
                }
                var createTable         = function(){
                        var qrcode      = new QRCode(options.typeNumber,
options.correctLevel);
                        qrcode.addData(options.text);
                        qrcode.make();
                        var $table      = $('<table></table>')
                                .css("width", options.width+"px")
                                .css("height", options.height+"px")
                                .css("border", "0px")
                                .css("border-collapse", "collapse")
                                .css('background-color', options.background);
                        var tileW       = options.width / qrcode.getModuleCount();
                        var tileH       = options.height / qrcode.getModuleCount();
                        for(var row = 0; row < qrcode.getModuleCount(); row++ ){
                                var $row = $('<tr></tr>').css('height',
tileH+"px").appendTo($table);

                                for(var col = 0; col < qrcode.getModuleCount(); col++ ){
                                        $('<td></td>')
                                                .css('width', tileW+"px")
                                                .css('background-color', qrcode.isDark(row, col)
? options.foreground : options.background)
                                                .appendTo($row);
                                }
                        }
                        return $table;
                }
                return this.each(function(){
                        var element     = options.render == "canvas" ? createCanvas() :
createTable();
                        $(element).appendTo(this);
                });
        };
})( jQuery );
```

Warnings

Unfortunately I used an open source library which ain't defined in Closure Compiler. That's why it shows so many warnings. The closure compiler doesn't know about the methods which are defined in the library. QRCode generator is a nice library for creating security, artificialy.

All the following warnigs are about the methods he didn't find in the known libraries. So I can say the code is damn good, and optimized.

Number of warnings: 23

```
JSC_INEXISTENT_PROPERTY: Property fn never defined on $ at line 2 character 0
$.fn.qrcode = function (options) {

JSC_INEXISTENT_PROPERTY: Property H never defined on QRErrorCorrectLevel at line 15
character 15
correctLevel : QRErrorCorrectLevel.H,

JSC_INEXISTENT_PROPERTY: Property addData never defined on qrcode at line 23 character
0
qrcode.addData(options.text);

JSC_INEXISTENT_PROPERTY: Property make never defined on qrcode at line 24 character 0
qrcode.make();

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 33
character 29
var tileW = options.width / qrcode.getModuleCount();

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 34
character 29
var tileH = options.height / qrcode.getModuleCount();

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 37
character 24
for( var row = 0; row < qrcode.getModuleCount(); row++ ){

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 38
character 24
for( var col = 0; col < qrcode.getModuleCount(); col++ ){

JSC_INEXISTENT_PROPERTY: Property isDark never defined on qrcode at line 39 character
16
ctx.fillStyle = qrcode.isDark(row, col) ? options.foreground : options.backgr...

JSC_INEXISTENT_PROPERTY: Property addData never defined on qrcode at line 53 character
0
qrcode.addData(options.text);

JSC_INEXISTENT_PROPERTY: Property make never defined on qrcode at line 54 character 0
qrcode.make();

JSC_INEXISTENT_PROPERTY: Property css never defined on ? at line 57 character 13
var $table = $('<table></table>')

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 65
character 28
var tileW = options.width / qrcode.getModuleCount();

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 66
```

```
character 29
var tileH = options.height / qrcode.getModuleCount();

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 69
character 23
for(var row = 0; row < qrcode.getModuleCount(); row++ ){

JSC_INEXISTENT_PROPERTY: Property appendTo never defined on ? at line 70 character 11
var $row = $('<tr></tr>').css('height', tileH+"px").appendTo($table);

JSC_INEXISTENT_PROPERTY: Property css never defined on ? at line 70 character 11
var $row = $('<tr></tr>').css('height', tileH+"px").appendTo($table);

JSC_INEXISTENT_PROPERTY: Property getModuleCount never defined on qrcode at line 72
character 23
for(var col = 0; col < qrcode.getModuleCount(); col++ ){

JSC_INEXISTENT_PROPERTY: Property appendTo never defined on ? at line 73 character 0
$('<td></td>')

JSC_INEXISTENT_PROPERTY: Property css never defined on ? at line 73 character 0
$('<td></td>')

JSC_INEXISTENT_PROPERTY: Property isDark never defined on qrcode at line 75 character
25
.css('background-color', qrcode.isDark(row, col) ? options.foreground : optio...

JSC_INEXISTENT_PROPERTY: Property each never defined on this at line 84 character 7
return this.each(function(){

JSC_INEXISTENT_PROPERTY: Property appendTo never defined on ? at line 86 character 0
$(element).appendTo(this);
```

Result

```
// Input 0 var f = jQuery; f.m.n = function(a) { "string" === typeof a && (a =
{text:a}); a = f.extend({}, {j:"canvas", width:256, height:256, f:-1,
d:QRErrorCorrectLevel.k, background:"#ffffff", e:"#000000"}, a); return
this.l(function() { var b; if("canvas" == a.j) { b = new QRCode(a.f, a.d);
b.g(a.text); b.i(); var g = document.createElement("canvas"); g.width = a.width;
g.height = a.height; for(var j = g.getContext("2d"), e = a.width / b.b(), h = a.height
/ b.b(), d = 0;d < b.b();d++) { for(var c = 0;c < b.b();c++) { j.fillStyle = b.h(d, c)
? a.e : a.background, j.fillRect(Math.round(c * e), Math.round(d * h), Math.ceil((c +
1) * e) - Math.floor(c * e), Math.ceil((d + 1) * e) - Math.floor(d * e)) } } }else { b
= new QRCode(a.f, a.d); b.g(a.text); b.i(); g = f("<table></table>").a("width",
a.width + "px").a("height", a.height + "px").a("border", "0px").a("border-collapse",
"collapse").a("background-color", a.background); j = a.width / b.b(); e = a.height /
b.b(); for(h = 0;h < b.b();h++) { d = f("<tr></tr>").a("height", e + "px").c(g); for(c
= 0;c < b.b();c++) { f("<td></td>").a("width", j + "px").a("background-color", b.h(h,
c) ? a.e : a.background).c(d) } } } b = g; f(b).c(this) }) };
```

2. Refactoring

As you can understand I can't do anything in the resulted code after my Closure Compile tool gave me, so I will refactor the initial code.

As long as the tool didn't find any heavy mistakes, just some like there should be less space on some lines and this ain't hard to correct.

The next step in refactoring would be to make the code more clear, for someone else who see this code to understand it, it should have more sugestive names. Maybe the code will more heavily in size but it will be more readable. Also adding comments to code help to improve the quality of code. The white spaces between lines also counts. Because it separates the code visualy in blocks of code, also separating the functionalities and compartiments of functions.

Here is the result:

```
(function( $ ){
	$.fn.qrcode = function(options) {
		// if options is string,
		if( typeof options === 'string' ){
			options		= { text: options };
		}

		// set default values
		// typeNumber < 1 for automatic calculation
		options		= $.extend( {}, {
			render : "canvas",
			width : 256,
			height : 256,
			typeNumber : -1,
			correctLevel : QRErrorCorrectLevel.H,
	background	: "#ffffff",
	foreground	: "#000000"
		}, options);

		var createCanvas = function(){
			// create the qrcode itself
			var qrcode = new QRCode(options.typeNumber,
options.correctLevel);
			qrcode.addData(options.text); //place where we add the input text
			qrcode.make();//generating the image

			// create canvas element
			var canvas = document.createElement('canvas');
			canvas.width = options.width;		//adding the corresponding size to
the images.
			canvas.height = options.height;
```

```
                        var ctx = canvas.getContext('2d'); //standard function to create a
image using .canvas method

                        // compute tileW/tileH based on options.width/options.height
                        var tileW = options.width  / qrcode.getModuleCount();
                        var tileH = options.height / qrcode.getModuleCount();

                        // draw in the canvas
                        //fullfiling the image with encrypted data
                        for( var row = 0; row < qrcode.getModuleCount(); row++ ){
                                for( var col = 0; col < qrcode.getModuleCount(); col++ ){
                                        ctx.fillStyle = qrcode.isDark(row, col) ?
options.foreground : options.background;
                                        var w = (Math.ceil((col+1)*tileW) -
Math.floor(col*tileW));
                                        var h = (Math.ceil((row+1)*tileW) -
Math.floor(row*tileW));

        ctx.fillRect(Math.round(col*tileW),Math.round(row*tileH), w, h);
                                        }
                        }
                        // return just built canvas
                        return canvas;
                }

                // from Jon-Carlos Rivera (https://github.com/imbcmdth)
                // create the qr code using table method
                var createTable         = function(){
                        // create the qrcode itself
                        var qrcode = new QRCode(options.typeNumber,
options.correctLevel);
                        qrcode.addData(options.text); //adding the encrypted text which will
be later encrypted into the image
                        qrcode.make();

                        // create table element
                        var $table = $('<table></table>')
                                .css("width", options.width+"px")
                                .css("height", options.height+"px")
                                .css("border", "0px")
                                .css("border-collapse", "collapse")
                                .css('background-color', options.background);

                        // compute tileS percentage
                        var tileW = options.width / qrcode.getModuleCount();
                        var tileH = options.height / qrcode.getModuleCount();
```

```
                        // draw in the table
                        for(var row = 0; row < qrcode.getModuleCount(); row++ ){
                                var $row = $('<tr></tr>').css('height',
tileH+"px").appendTo($table);

                                for(var col = 0; col < qrcode.getModuleCount(); col++ ){
                                        $('<td></td>')
                                                .css('width', tileW+"px")
                                                .css('background-color', qrcode.isDark(row, col)
? options.foreground : options.background)
                                                .appendTo($row);
                                }
                        }
                        // return just built canvas
                        return $table;
                }


                return this.each(function(){
                        var element    = options.render == "canvas" ? createCanvas() :
createTable();
                        $(element).appendTo(this);
                });
        };
})( jQuery );
```

Refactoring have improved code quality, made it way more readable. The comments added to the code made it really clear and easy to read. I guess anybody could understand this code.

3. Architectural Patterns

An **architectural pattern** is a standard design in the field of software architecture. The concept of an architectural pattern has a broader scope than the concept of design pattern. The architectural patterns address various issues in software engineering, such as computer hardware performance limitations, high availability and minimization of a business risk. Some architectural patterns have been implemented within software frameworks.

3.1 ETL (Extract, Transform and Load)

Extract, Transform and Load (**ETL**) refers to a process in database usage and especially in data warehousing that involves:

- Extracting data from outside sources
- Transforming it to fit operational needs (which can include quality levels)
- Loading it into the end target (database, more specifically, operational data store, data mart or data warehouse)

As an example we will use a database with QRCode Images, in which will be included the names of all students from a certain group. So every student will have their own personalized QRImage, and this image will be printed on the test so that test will not be replaced. This feature will improve the security.

We need to create the application this way that QRCodes will be generated from an external database which will contain the names of all students. Every field in the table will correspond to a certain student.

The application will communicate with the precreated Database, and will generate automatically the images. The only need in teacher's activity will be to fullfill the database with necccessary names, and their group, but once, the database will be created it won't have the necessity to alter it.

So as we can see our application correspond to the principle of SOA.

- It extracts the data from a database.
- Application transform all the data into operational data and its needs.
- All the data is transferred to the end target application layer, and the data doesn't need to be refactored.
- So, in the end of the process we will obtain a well-structured document with all necessary information.

3.2 MFT (Managed File Transfer)

Managed file transfer (MFT) refers to software solutions that facilitate the secure transfer of data, in flight and at rest, from one computer to another through a network (e.g., the Internet). MFT solutions are often built to support the FTP network protocol. However, the term specifically describes solutions that remedy the disadvantages associated with FTP.

Typically, MFT offers a higher level of security and control than FTP. Features include reporting (e.g., notification of successful file transfers), non-repudiation, auditability, global visibility, automation of file transfer-related activities and processes, end-to-end security, and performance metrics/monitoring.

MFT applications are available as both licensed software packages and SaaS solutions. Some are specially designed for enterprise use while others are for sale to individual consumers. A few enterprise-focused SaaS MFT providers also manage the additions of new trading partners, which can free up a lot of IT resources.

In our application MFT will be implemented on printing complete tests.

Tests can be generated on any computer, and transferred to a local printer in the university. This way application is really flexible and universal.

Our application will have an extraordinary feature. Largely used last years in software parctice and science, cloud storage. The application will automatically upload all the test to a local UTM server where the tests could be verified by authorized persons. For example the Dean can access any time the files to see the tests and verify if the tests correspond to the standards. The files will be transferred over the server over File Transfer Protocol, or HTTPS.

If the professor has a computer home with acces to internet he can easily make the test at home in comfort with a cup of coffee or tea, without bothering about some problems that the files could be lost.

The goals of this architectural pattern are:

- The application support multiple file transfer protocols (FTP, HTTPS)
- Securely transfer files over public and private networks using encrypted file transfer protocols. HTTPS and FTP were designed to be secured initially so we don't need any other further development.
- All the generated data will be stored on the utm server.
- HTTPS and FTP has built-in detection and handling of failed file transfers. At the end application shows the log in which there are all the results.

3.3 EAI(Enterprise Application Integration)

Enterprise application integration (EAI) is the use of software and computer systems architectural principles to integrate a set of enterprise computer applications.

At the level of our application we can integrate it by sharing it to every teacher in the university.

Regarding the cloud space, every teacher can have some limited space on the server, for example 5GB, this will be sufficient. All the data is kept there and there is no need to have radnom trash on your computer. The server can set that data older than 18month to be automatically deleted. And free space is automatically gained.

3.4 ESB (Enterprise Service Bus)

An enterprise service bus (ESB) is a software architecture model used for designing and implementing the interaction and communication between mutually interacting software applications in service-oriented architecture (SOA). As a software architecture model for distributed computing it is a specialty variant of the more general client server software architecture model and promotes agility and flexibility with regards to communication and interaction between applications. Its primary use is in enterprise application integration (EAI) of heterogeneous and complex landscapes.

In our SOA application there will be implemented some ESB duties will need to be specified. So, as interacting software applications will serve the main device itself with basic application and the print device or the server will be placed the information.
Some of the duties duties of an ESB will be :

- Monitor and control routing of message exchange between services
- Resolve contention between communicating service components
- Control deployment and versioning of services
- Cater for commodity services like event handling, data transformation and mapping, message and event queuing and sequencing, security or exception handling, protocol conversion and enforcing proper quality of communication service.

ESB will have to transform the message into a format that the application can interpret. A software “adapter” fulfills the task of effecting these transformations (analogously to a physical adapter).

Bibliography

[1] Code analysis tools.

http://closure-compiler.appspot.com/home

[2] Description of the Closure Compiler

https://developers.google.com/closure/compiler/

[3] The Architectural patterns and their description

http://en.wikipedia.org/wiki/Architectural_pattern

[4] ETL pattern, objective and purpose.

http://en.wikipedia.org/wiki/Extract,_transform,_load

[5] MFT pattern, objective and purpose.

http://en.wikipedia.org/wiki/Managed_File_Transfer

[6] ESB pattern, objective and purpose.

http://en.wikipedia.org/wiki/Enterprise_service_bus