GEEKY.GENT

P O R T F O L I O

"HELLO WORLD!”

THE HEAD BEHIND GEEKY.GENT & GIT.COACH

I’m Thomas, a 30 year old indie developer and IT consultant from Paderborn, Germany.

Over the last couple of years, I’ve worked on — and contributed to — a wide variety of projects. This portfolio is going to focus on the ones I have something interesting to tell about: The ones with peculiar stories, unique challenges and deep impacts.

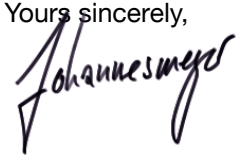
Different projects, which brought with them not only a conglomeration of languages and therefore patterns, but also required diverse approaches: While the deployment of a network-printer driver in C has its focus on efficient data compression, a reactive UI element obtains its core values from reusabilty, configurability and well documented code for human readability.

For my own projects, I have been responsible for every single step of the development process: Design, architecture, quality control, icon design, web design, marketing, video editing, finance management, license control, etc. pp. Software architecture, user interface design and version control stand out as my passions here.

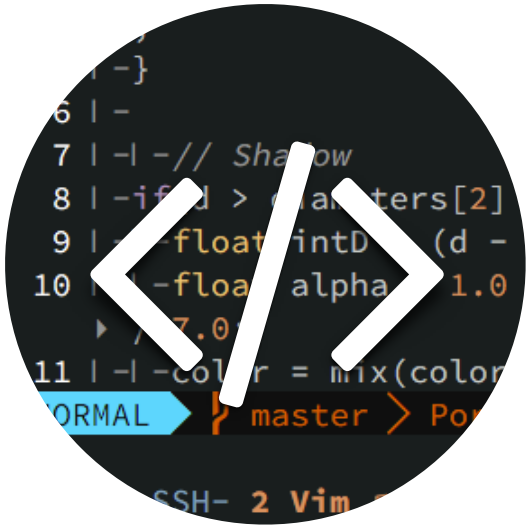
This document is designed for digital consumption. It contains hyperlinks.

Thank you for your interest and time. Enjoy…

Yours sincerely,



TOOLS



VIM & EMACS I prefer Emacs as my personal editor. I switched from Vim and have configured both editors to behave identically. My config files have grown over the past couple of years and contain several thousand lines of custom setup. I invite you to take a peak.

GIT I use Git as my main version control and backup system. It is superbly integrated into Vim & Emacs, alternatively I prefer the raw commandline version. At the office I run three redundant backup solutions two of which are based on Git. I also teach it at Git.Coach

SKETCH DESIGN I use Sketch to layout all of my user interfaces and most other designs. Every icon of my applications is designed by me. If you compare old icons like "Chain The Arc!" to newer ones like "*SNIP*" my progress as a self taught designer is quite noticable.

CONSULTANT / SPEAKER / WRITER

07

TUTORSHIP

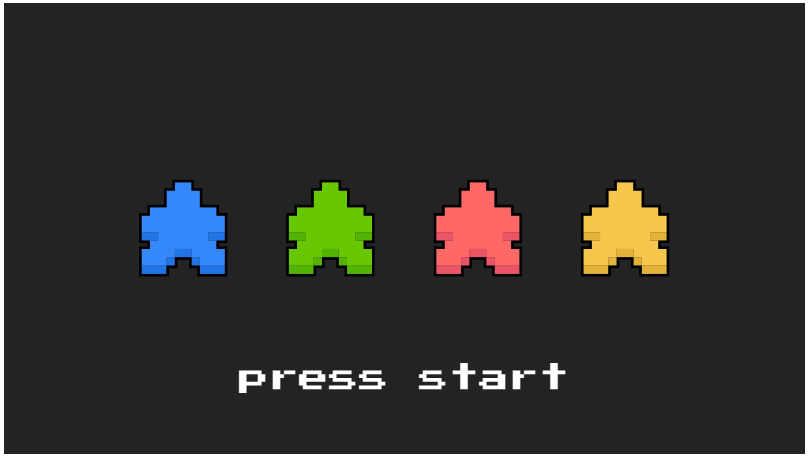
AT UNIVERSITY

11

CONSULTANT &

GIT COACHING

UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

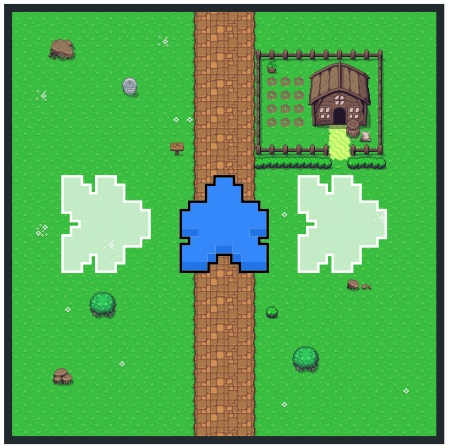


TUTORSHIP

SOFTWARE ENGINEERING

During the course of a semester student groups of 10 to 15 members create a multiplatform (PC & Android) multiplayer board game and an AI. Students are taught **scrum**, **project management**, **version control**, **DevOps** and of course **software development**. Tutoring the groups was a challenging task, especially from the interpersonal perspective: Conflicts had to be solved and constructive criticism had to be provided.

The tutorship got me into teaching version control...



Animated Carcassonne tile

Timeframe: **August '18 - February '19**

Website: **geeky.gent/woc/**

Video-Demo of four fuzzy neural nets playing Carcassonne:

youtube.com/watch?v=ZaC-NGlGhkM

Languages: **Java**

GIT.COACH

CONSULTING, WORKSHOPS, TALKS

Prior to my engagement as a tutor version control was not part of the curriculum of the Software Engineering course. The introduction presentation had — quite literally — five slides regarding Git and everybody was expected to *just know it*. I started giving Git workshops for the participants and they were received incredibly well. So well infact, that I got motivated to do this professionally. I was hired to hold workshops for the students and therefore able to refine my teaching material over the course of two years.

At the moment (early 2020) I have three courses in my repertoire, while working on two others and a book.

If you are interested in more details like the approach of the courses and my philosophy behind them, feel free to visit my website: **www.git.coach**.

Timeframe: **August '18 - NOW**

Website: **git.coach**

Tools: **Docker, Gitlab, org-mode**

Frameworks: **HUGO, RevealJS**

COMMERCIAL PROJECTS

08

CONDENSE

10

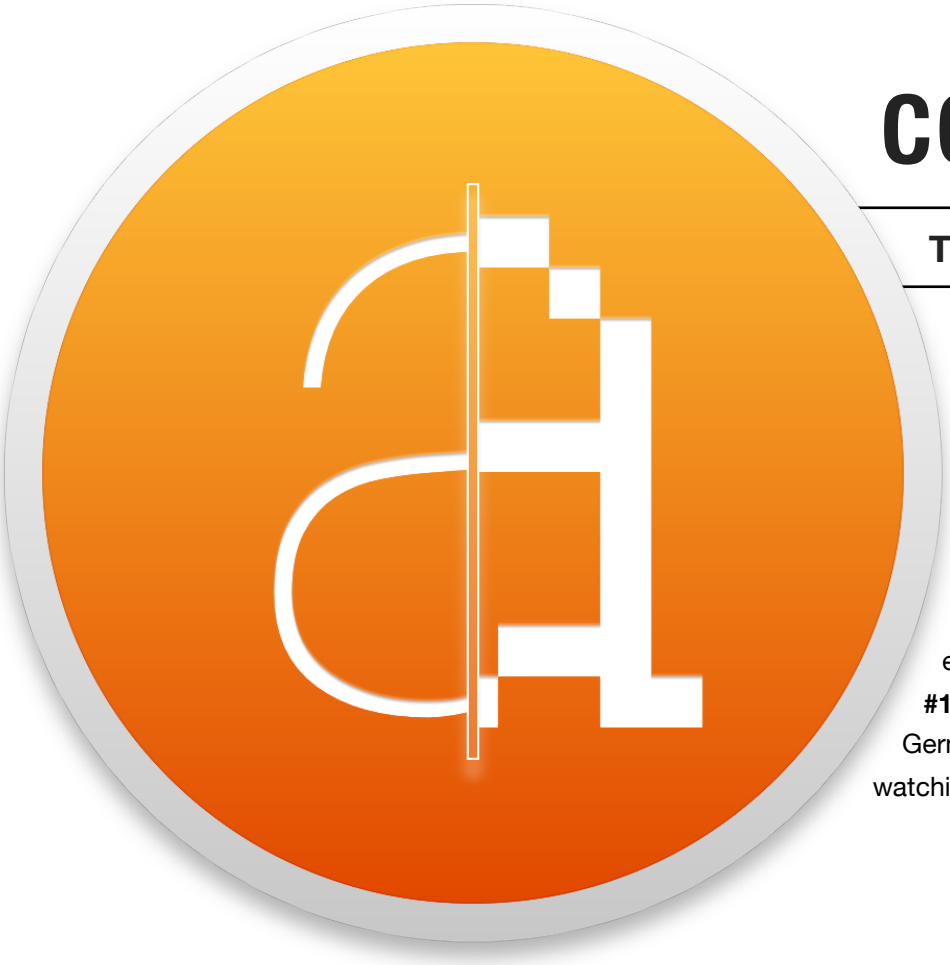
MOUSEBEAM

12

CHAIN THE ARC

14

SNIP



CONDENSE

THE NEW OCR: SIMPLE, FAST & ELEGANT

Condense has been *the* application which allowed me to quit my job and follow my own path as an independent software developer. It started as a hobby project while studying at university: We were oftentimes handed lecture-scripts as protected PDFs to prevent copyright infringements. Since I was tired of typing out notes and I did not find an elegant OCR solution to my problem, I wrote this app by and for myself. It became very popular amongst my circle of friends / university colleagues who encouraged me to publish it on Apple's AppStore. **Condense became the #1 Top Selling macOS application** in several European countries including Germany. Since images say more than a thousand words I highly recommend watching the video-demo linked below.

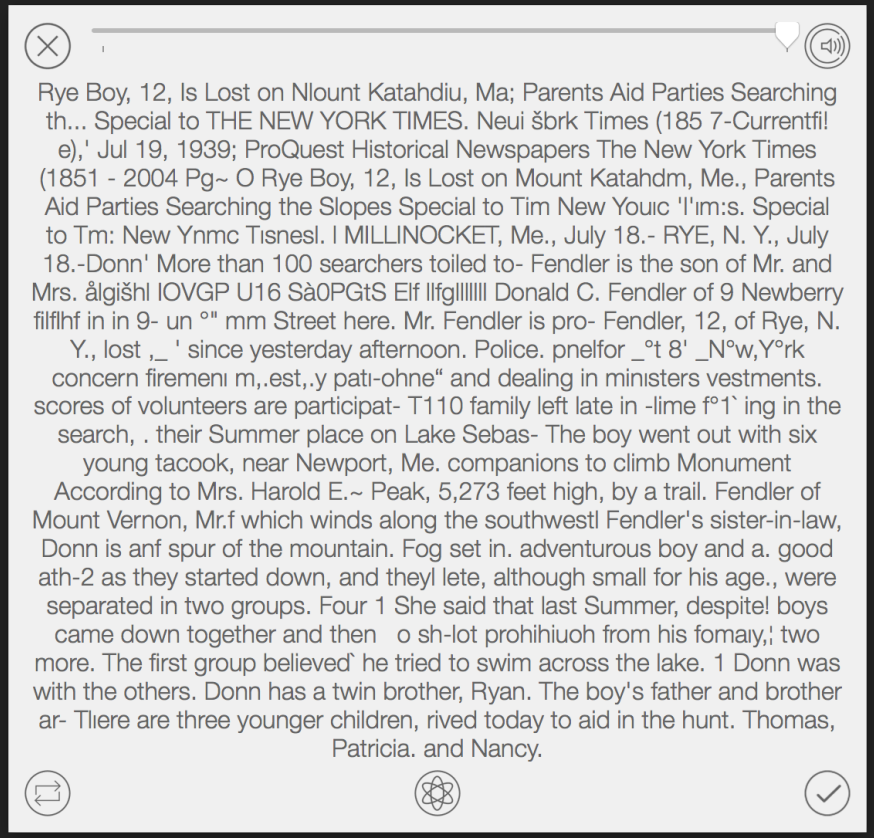
Website: www.condenseapp.com **(offline)**

Video-Demo: youtube.com/watch?v=enWml__qVVY

Languages: **Objective-C, C++**

Availability: **2014 - 2019 Mac AppStore**

CHALLENGES



Resulting “OCR Gibberish” when scanning a typeface of size 12

Data handling and distillation

Condense uses the **Tesseract OCR** framework which was originally created by HP and later acquired and open sourced by Google. Its purpose was print recognition of >300dpi scans. If you throw a 12point typeface at this framework, you won't extract anything, but random gibberish. A big challenge in development was to prepare the extracted image data for Tesseract. Images are scaled up by factor four per side for example. This means that extracting a 100x100px field will result in a 400x400 = 160,000 pixel image. Handling these copious amounts of data requires me to process some of the algorithms on the GPU.

Following the extraction the resulting string is usually filled with seemingly random characters and demands reexamination. Condense uses some custom algorithms concluding assumptions about the underlying text, using Hamming distances and dictionaries to generate a usable result.

This multi-step data extraction reminded me of a distillation process. That's how Condense received its name.

MOUSEBEAM

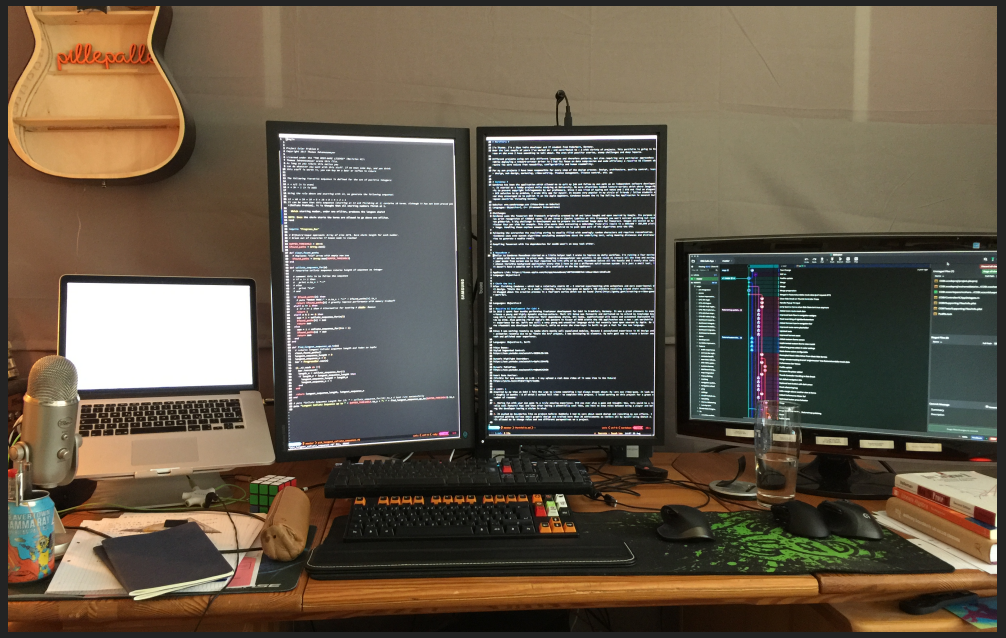
4M: MULTI-MONITOR-MOUSE-MADNESS

Similar to Condense MouseBeam started as a little helper tool I wrote to improve my daily workflow. I'm running a four monitor setup with two screens in pivot mode. Managing a mousepointer was painful: It got stuck at corners all the time and moving your mouse from screen one to four required two lift offs of my arm. MouseBeam solves all the hassle and I truly miss this almost invisible background application every time I have to use a different multi-screen system. It's just a small tool, it doesn't have a website nor a trailer. It's available in the Mac AppStore.

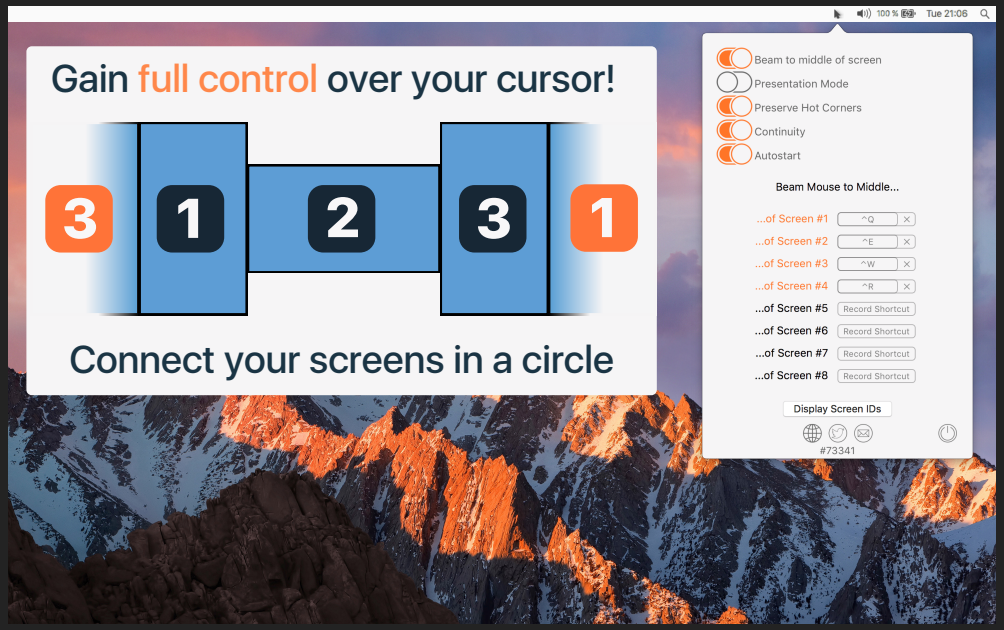
Language: **Objective-C**

Availability: **2017 - 2019 Mac AppStore**

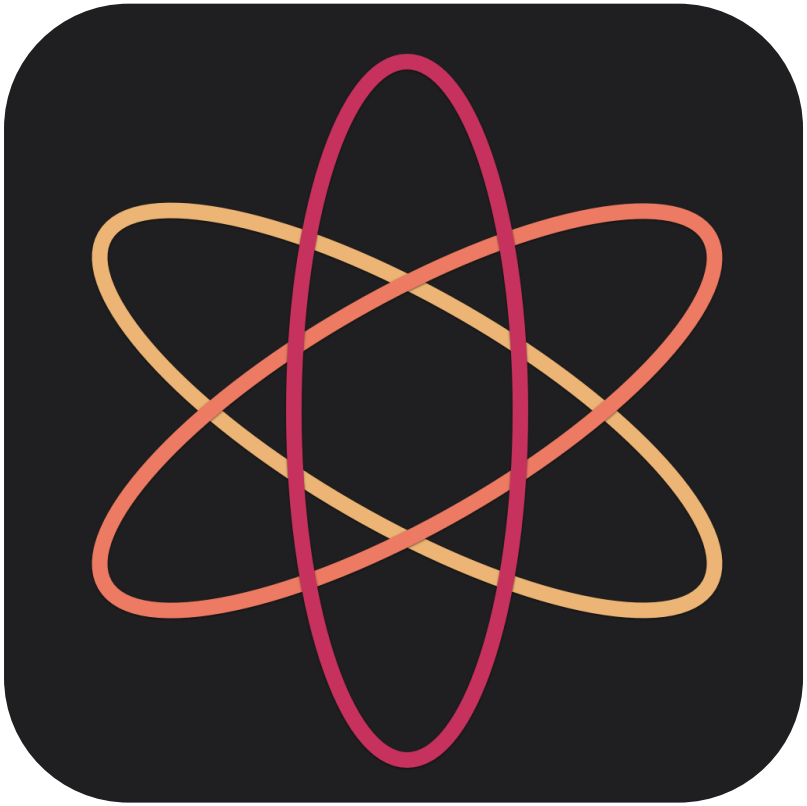
PROBLEM / SOLUTION



Workstation — Using your mouse will become a daily annoyance here



MouseBeam completely changes your workflow and has become a beloved app amongst video editors and developers alike



CHAIN THE ARC

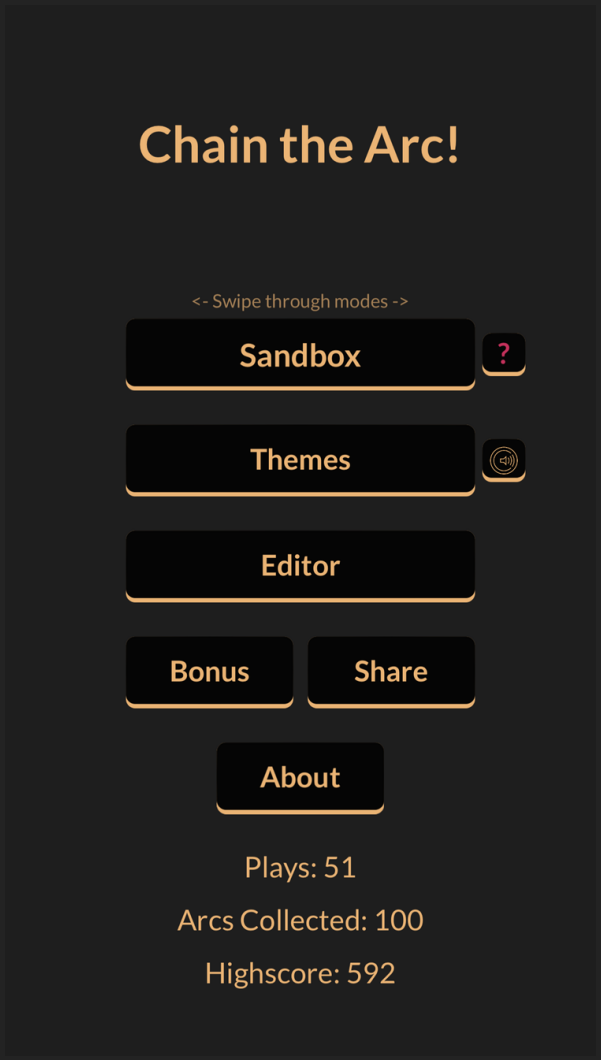
ADDICTIVE LIKE BUBBLEWRAP

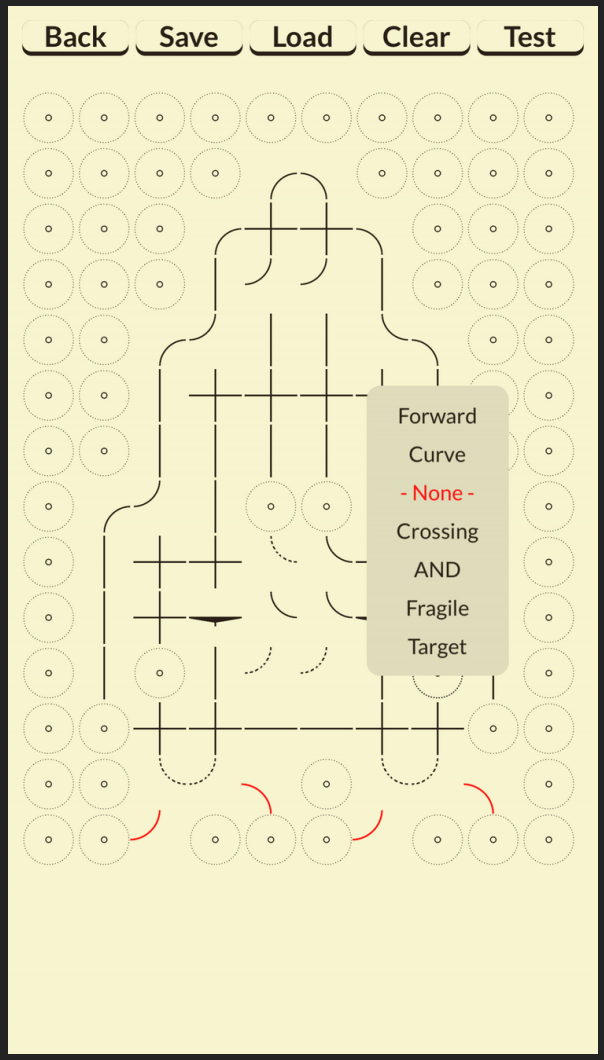
After finishing Condense — which had a relatively static UI — I started experimenting with animations and more experimental UI design. "Chain the Arc" is a small, relaxing, free-to-play-game on Apple's iOS platform revolving around chain reactions. I blogged about its creation process in a four-part series which can be found on my blog.

Language: **Objective-C**

Availability: **2016 - 2019 Mac AppStore**

SCREENSHOTS





Screenshots taken from the game

SNIP

“[...]CHALLENGING INDIE GAME PEARL."

Inspired by my freelance time at Cobi (explained later), I felt the urge to create something I had always dreamt about: My very own video-game. It took me roughly 14 months — 8 of which I worked full time — to complete this project. I had to overcome a considerable amount of obstacles and challenges to create this game. As a one-man studio I do not possess the manpower to develop a story driven game with hand made stages and characters. Snip is a highscore based game and its content is generated procedurally. The goal of the game is to master snipping a disc precisely.

As a mobile game, it will fit into your lunchbreak — a playthrough rarely takes more than 5 minutes. You can challenge your friends using Game-Center, or by sharing your score on social networks.

Website: www.snip.rocks **(Includes Game Trailer)**

Languages: **Objective-C, C for GPU-Shaders,**
JavaScript / CSS (Website)

Video-Demo: **Animated UI:** youtube.com/watch?v=8k-aLItmOP4

Availability: **2016 - 2019 iOS AppStore**

GRAPHIC DESIGN

Marketing banner created for *SNIP* using Sketch 3

Some of the over 30 achievements I designed for *SNIP* — Each made up of 5 vector layers and colorized programmatically within the game

Main Menu

The User Interface is completely custom made to reflect the game's theme in multiple ways: Buttons are shaped like discs and -are animated smoothly. The colourscheme for the whole game gets parsed from a file. A reskin for a special event like Christmas would be done within less than 5 minutes while keeping a uniform style. Demo

Particle Effects / Colour Blind Mode

The confetti- / dust-particle effects are generated on the fly using the current mode's scheme. To make the game more accessible for handicapped users, distict shapes are displayed on top of the obstacles.

Gameplay Variety

The game offers 16 very unique disc types. Every disc type is usable in every of the 3 different game modes offering different benefits to the player. Picking the right discs for the job and swapping on the fly brings strategical depth to an action game.

Pop cultural- / historic-references wrapped in achievements

I loved working on this project for a number of reasons:

1. **Having fun with your own game is a truly amazing experience.** Did you ever play a game and thought: Wow, this would be a really cool feature? Yep! And then after having a potentially cool idea, you have to swap roles between being a player and being the developer having a vision in mind.

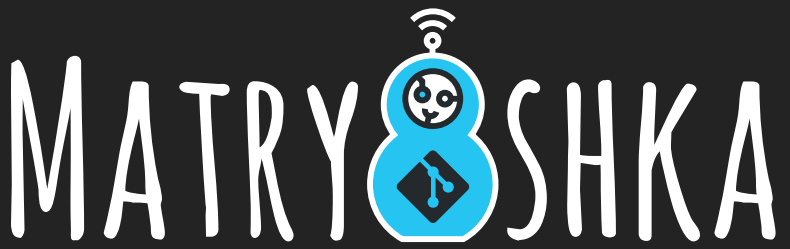
2. **It pushed my boundaries like no project before:** Suddenly, I had to care about sound design and recording my own effects. I started getting serious about graphic design and crafted more than 30 achievements as vectors all by myself using Sketch 3. It allowed me to change roles and see different perspectives on a project.

3. **Code optimization played a huge role in the development process.** Let's be honest here: Usually you don't get paid to create a solution which is a bit faster and more efficient because it simply isn't worth your time as a developer compared to the pay off you get *in a standard application*. In a game however — where physics and graphics are calculated 60 times a second — you better become as efficient as possible. Managing heap- / stack-allocations and keeping a pool for recycled game objects were very challenging and interesting tasks.

4. **Code design with creative flexibility and extensibility in mind:** I designed *SNIP* *while* I was developing its codebase. A new game mode, a new disc type or gameplay changes could appear at any time. A lot of thought was put into balancing feature bloat and extensibility. The engine is able to slow down time for example while still maintaining accurate physic- / collision-calculations.

OPENSOURCE PROJECTS

19
TAGSNAG



20
MATRYOSHKA

TAGSNAG (OPENSOURCE)

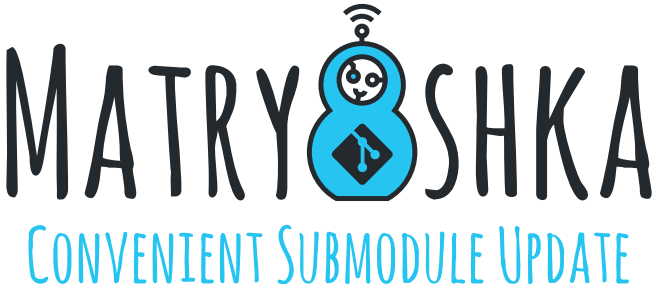
EDUCATIONAL GIT MANAGEMENT

During the winter term 2018/2019 I tutored one of our university's computer science courses called Softwaretechnik-Praktikum (Software Engineering) and realized that management of multiple Git repositories — oftentimes organized by *inexperienced* students — is **not a trivial task** at all. Contrary to usual Git workflows where every repository is quite unique and has to be managed individually, educational work and management oftentimes requires us to run a **set of commands** over ***multiple*, rather similary organized repositories**. These commands shall neither be invasive nor destructive, so even if the administrator sat down and developed a series of bash scripts, the developer of such a script would be forced to deploy defensive counter measures and thoroughly test this little script before using it in a live production environment.

Inspired by this set of problems I developed an open source CLI tool called **TagSnag**. It is written in Python 3 and therefore platform independent. The current version has been (manually) tested on macOS, Linux and Windows 10. It enabled my colleagues and myself to update all repositories at once and also extract files & folders from a provided git-tag. The project has proven its usefulness and passed the alpha status. For more information about the project I suggest visiting its official **Github** repository. The readme provides detailed insight into installation and usage.

Github: **/t89/tagsnag**

Languages: **Python 3**



MATRYOSHKA (OPENSOURCE)

CONVENIENT SUBMODULE UPDATE

While working on the "Dependency Analysis Webservice" mentioned earlier, I was responsible for managing the Git repositories with a wide variety of submodules. Most of them were in active development and a pain to organize and keep updated within multiple repositories.

I wrote a small and well documented set of bash scripts to automize the complete process and make it more convenient. The tool is fairly popular amongst former colleagues at the chair of Secure Software Engineering.

Github: /t89/matryoshka

Languages: Bash

FREELANCE / CONTRACT WORK

CHAIR OF SECURE SOFTWARE ENGINEERING

SOOT SUITE - STATIC SOFTWARE ANALYSIS

When code is compiled into bytecode it's optimized greatly. Optimizations might be: Different loop types are compiled into one type, switch statements behave differently depending on the switchable type and variable names are omitted completely.

If you translate this bytecode back into a meta language (Jimple in this case)— which looks like a mixture of C and Assembler — you are able to gain incredibly interesting insights into your program.

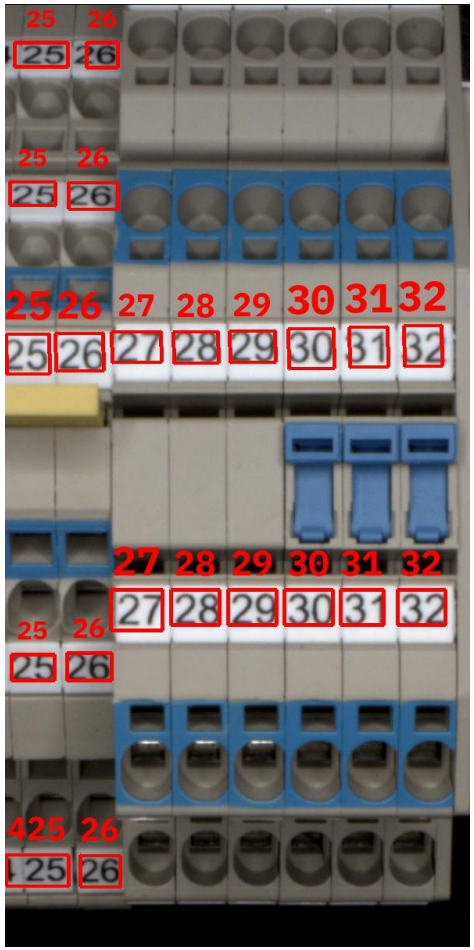
You can check for rebundled code (the same code in different bundles), precisely tell if you are affected by a security issue causing method and analyse your code on a deep level.

I joined an unreleased soot *related* project. Tasks ranged from writing unit tests to implementing the data structures responsible for handling the Java switch statements.

Timeframe: **April '19 - August '19**

Languages: **Java, Jimple**

Frameworks: **Soot**



OCR Visualization

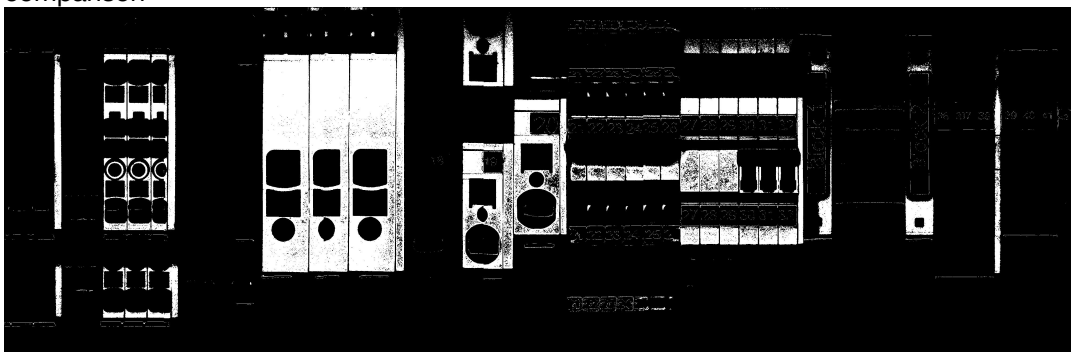
Weidmüller

OPTICAL COMPONENT COMPARISON

Extracting relevant data from an enormous data set is challenging. Especially so, if the data set is a stream of high definition images, taken by multiple cameras from different angles.

I'm working on a system capable of analyzing electronic components which are then compared to a golden-master component. The master is labeled and configured correctly. The system notifies the personal when it detects possible misconfigurations and displays the area on a display.

Challenges: Handling gigabytes of data, OCR, color coding, shape detection, tuning tolerance values, image pre-processing, creating unified data structures for efficient comparison



Generated mask used for shape recognition

Timeframe: **March '20 - now**

Languages: **Python**

Frameworks / APIs: **Tesseract, GoogleCloudAPI, OpenCV, numpy, Tensorflow**

CHAIR OF SECURE SOFTWARE ENGINEERING

DEPENDENCY ANALYSIS WEBSERVICE

In collaboration with a PhD. Student at the University of Paderborn we initiated the development of a dependency analysis website consisting of frontend, backend, and a sophisticated data model shared by both. The model was written in Kotlin to keep the classes more compact.

Tasks included...

...designing and developing the data model

...developing a distributed, dockerized, python 3 (scrapy) webcrawler hosted on AWS to parse and analyze the data of ~4.8 million packages of maven central. This data could be funneled into the REST backend via tokens

...repository management of 3 interlinked repositories using submodules and Gitlab CI

Timeframe: **April '19 - August '19**

Languages: **Java, Kotlin, Python 3, Bash**

Tools: **Docker, AWS, Gitlab CI**

Frameworks: **Spring Boot, Soot, Scrapy**

COBI (FREELANCE)

REACTIVE UI DEVELOPMENT

In 2015, I spent four months doing freelancer development for the Cobi GmbH in Frankfurt, Germany. It was a great pleasure to experience a young and highly dynamic startup-culture. Their Git setup was fantastic and motivated me to extend my knowledge beyond Git's standard version control features: Their dependency chains, Git hooks, sophisticated unit tests and automated deployment blew my mind. Also getting rid of Apple's MVC pattern in favour of MVVM within iOS development was a new — yet worthwhile and enlightening — experience for me. *Swift* was still a very young and unsettled language, which had not been open sourced by Apple yet. So the viewmodel was developed in Objective-C, while we wrote the view-layer in *Swift* to get a feel for the new language.

Since I was working remotely, my tasks where mostly well capsulated modules. Because I accumulated experience in UI design and animation recently due to my "Chain the Arc" project, I was developing UI elements. My main goal was to create a butter smooth and polished user experience.

Website: **www.cobi.bike**

Languages: **Objective-C, Swift**

UI-Demos: **Styled Segmented Control**

Dynamic Highlight Searchbar

Dynamic TableView

Heart Rate Monitor **(visible for 2 seconds at 1:20)**

NEXOMA

DEVELOPER (2011-2014)

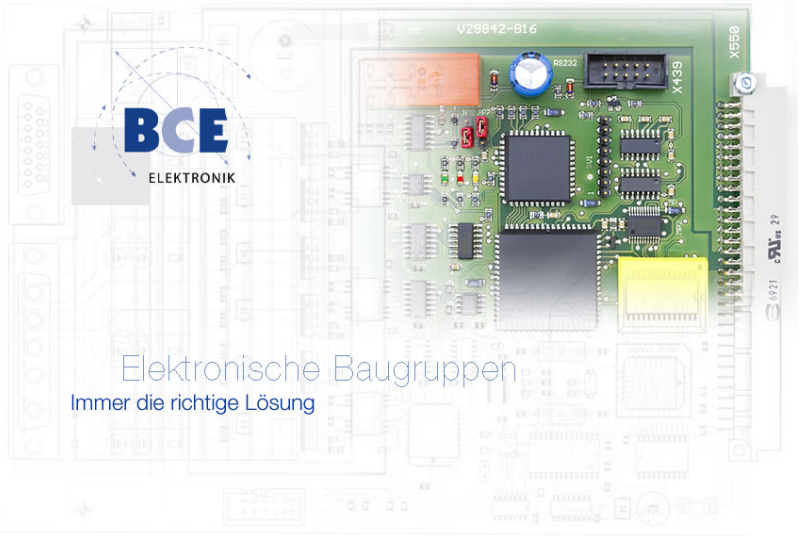
I worked for the Nexoma GmbH, located in Paderborn, where I developed several iOS- / macOS-apps and also supported their Android-Team by writing a handful of classes in Java. I took my first serious steps in iOS development here. We developed a course booking application for the VHS-Paderborn, an online shop integration for Hometrend and a magazine reader for Strobel.

I also got to experience CVS version control. If you consider Git complicated, I recommend trying CVS: It's as well documented as Mordor...

“It’s a dangerous business, Frodo, going out your door. You step onto the road, and if you don't keep your feet, there's no knowing where you might be swept off to.”

Website: **www.nexoma.de**

Languages: **Objective-C, Java**



BCE-ELEKTRONIK

EMBEDDED SYSTEMS DEVELOPER (2008-2010)

During my Computer Engineering studies, I was working for an engineering office located in Lemgo, Germany. My projects were developed using Windows XP, Pascal-Delphi, C, and some portions of Assembler.

Notable experiences:

- Developing for the german medical LDT (Labor-Datenträger) interface, which is fairly strict. Working with medical data in Germany is an interesting experience!
- Writing a minimalistic network printer driver.
- Programming an industrial sewing machine for car seats, which was manufactured for deployment in Italy / China. I was briefed that factory workers WILL try to misuse the apparatus on purpose to break it. Defensive programming / design was key.
- Subjecting data to suitable mathematical treatment and plotting it on a display: The raw data was taken from small electric engines and sensors of the Merlin Coagulometer. My job was to apply descriptive statistics and plot real time graphs.
- Designing simple PCBs using Eagle Software.
- Soldering / testing small badges of prototypes.

Website: **www.bce-elektronik.de**

Languages: **C, Pascal-Delphi, Assembler**

CONTACT

DIGITAL

WEB:	geeky.gent | git.coach
EMAIL:	contact@geeky.gent
GITHUB:	t89

ANALOG

Thomas Johannesmeyer
Holsteiner Weg 9b
33102 Paderborn

So long and thanks for all the fish.