

A rule-based geospatial reasoning system for trip price calculations



Stefan Schenk

Supervisor: Willem Brouwer

Advisor: Mewis Koeman

Department of Software Engineering
Amsterdam University of Applied Sciences

This dissertation is submitted for the degree of
Bachelor Software Engineering

March 2018

Todo list

<input type="checkbox"/> Write acknowledgement	ii
<input type="checkbox"/> Add chapter name	4
<input type="checkbox"/> Add pregame to appendix	4
<input type="checkbox"/> Refer to chapter containing methods and techniques about scrum	5
<input type="checkbox"/> appendix pregame ref	8
<input type="checkbox"/> Refer to figure	11

A rule-based geospatial reasoning system for trip price calculations

<i>Author</i>	<i>Stefan Schenk, 500600679, +31638329419</i>
<i>Place and date</i>	<i>Medemblik, 10 Mar 2018</i>
<i>Educational Institution</i>	<i>Amsterdam University of Applied Sciences</i>
<i>Department</i>	<i>HBO-ICT Software Engineering</i>
<i>Supervisor</i>	<i>Willem Brouwer</i>
<i>Company</i>	<i>taxiID, development team</i>
<i>Company address</i>	<i>Overleek 4</i> <i>1671 GD Medemblik</i> <i>Netherlands</i>
<i>Company Advisor</i>	<i>Mewis Koeman</i>
<i>Period</i>	<i>01 Feb 18 t/m 30 Jun 18</i>

kks32: Write acknowledgement

Abstract

A purely geometrical interpretation of user-defined locations would allow taxi-companies around the world to set up rules so that trip prices could be calculated without depending on distinct postal code systems. Geolocation datatypes provide part of the solution, but the benefits of geometrical definitions are lost when areas intersect. A precedence hierarchy of rules that are tied to reusable locations would eliminate these competing rule matches.

A solution is proposed to implement a microservice with a single responsibility of calculating trip prices that is accessible to existing systems and portals in which users can define the pricing rules. The company for which this system is realized requires customers to be able to migrate to the new system without downtime, while keeping the existing rules that determine the prices of taxi trips.

The core system manages user and company data, this complicates identity management in the microservice. A JSON Web Token would allow user identity to be stored in the payload of the token, thereby delegating authentication to the core system, maintaining the single responsibility of the microservice.

Table of contents

List of figures	vii
List of tables	ix
1 Introduction	1
1.1 Context	1
1.2 Assignment	2
1.3 Research	3
1.3.1 Questions	3
1.4 Process	4
2 Encoding Locations	7
2.1 Introduction	7
2.2 A Brief History Of Geographic Locations	7
2.3 Useful Location Types	8
2.4 Requisites of Locations	8
2.5 Literature Review	9
3 System Architecture	11
3.1 Introduction	11
3.2 Architectural Patterns	11
3.3 Authentication and Authorization	13
4 Proposed Solution	15
4.1 Introduction	15
4.2 Design	15
4.3 Methods and Techniques	15

5	Realization	17
5.1	What is loren ipsum? Title with math σ	17
5.2	Why do we use loren ipsum?	17
5.3	Where does it come from?	18
6	Conclusion	21
6.1	Short title	21
	References	27
	Appendix A How to install L ^A T _E X	29
	Appendix B Installing the CUED class file	33
	Index	35

List of figures

2.1	LatLngSphere	7
3.1	Architecture	12
6.1	Minion	22
6.2	Best Animations	26

List of tables

Chapter 1

Introduction

Automatic fare estimations and calculations are one of many common features to go hand in hand with taxi dispatch systems. A potential passenger wants to book a ride using a mobile app by selecting the pickup and drop off locations, so that the app can display a list of available vehicles and corresponding prices. The passenger chooses the desired vehicle from the list, and a taxi arrives to take the passenger to the destination. When the destination is reached, the system calculates the final price, with or without discounts, including taxes, and additional costs added by the driver. Based on whether the passenger ordered a ride from the center of Amsterdam, or a desolate hard to reach location, the cost of such a trip can be higher or lower. For this reason, companies must be able to define prices based on locations so that the delivered work is still profitable. The project for which this thesis is written aims to improve the process of defining prices based on various factors, one of which is geographical locations.

1.1 Context

The company for which the project is realised is taxiID, an Amsterdam based company providing end-to-end cloud solutions for taxi companies. Founded as a startup that successfully introduced smartphone taxi booking in The Netherlands, and offers a wide range of IT solutions to serve the taxi market, including a passenger app, a driver app, administrative panels, and track and trace hardware. taxiID solutions have proven to be a reliable set of tools for all size businesses. For independent taxi companies with 2 cars or a companies with large fleets, affordable solutions are available. taxiID's goal is to deliver affordable, time-saving solutions for taxi companies to allow for convenient planning and dispatching without requiring local installation. Though based in Amsterdam, the development team is located in Medemblik, consisting of two mobile developers, two backend developers, a de-

signer and two project managers. Clients are located across the globe, introducing challenges when developing applications that rely on clearly defined locations and infrastructures that vastly differ between countries. YourDriverApp (YDA) is a light version of the original solution that is offered by taxiID more focussed on smaller taxi companies. Currently this label depends on services within taxiID to operate, which has to change in the near future.

1.2 Assignment

YDA requires its own pricing calculation functionality that is similar to the existing taxiID implementation. All functionalities within the current system align with the clients demands, but some features introduce difficulties, for example: region names are too vague for specific database queries. A system must be implemented in which group admins can define pricing rules based on user defined locations and time schedules, that can be used for calculating a passengers trip price, or show prices of different products based on the trip the passenger is about to make. In the current system, locations are uploaded by taxi company group admins as excel sheets with departure and destination zip codes in conjunction with prices. This process works in countries with an unambiguous, explicit, well-defined postal code infrastructure. Postal codes are matched in efficient database queries, leaving less room for improvement in terms of performance. Interpretability is an issue however. Sheets may contain thousands of rows, making it hard to interpret and maintain. On top of that, countries without such systems are not covered by the functionality. There are two more types of pricing rules that cover the rest of trip pricing cases. A tier price system, that calculates fixed prices based cascading thresholds, and a dynamic pricing system that calculates prices per distance unit and minute. The term 'distance unit' is used on purpose, as distances are measured using different metrics in various countries. Pricing rules should be constrained by time frames, making rules available only for some hours a day, or only on christmas for example. Rules should be specifiable per product as different vehicle types have different prices, but are included in the same pricing rules. Discounts may be calculated with the trip price, and VAT should be displayed in the price breakdown. Some additional requirements to the system may be added in later phases, as Scrum is used to manage work iterations (this fact is covered later in this chapter). The system should be accessible to other systems, meaning that applications that currently rely on the old system should be able to migrate to the new system. As the old system shouldn't be used for new applications, as it was not designed for this use case. The system should have a single responsibility, and should be autonomous in that regard.

1.3 Research

Three main challenges that construct the assignment can be identified. Research must be done to attain the best possible way of mapping locations to pricing rules. What this means is that locations must be storable, comparable, and interpretable. The database must be able to store locations in an efficient manner, to which queries can be made as efficiently in order to find out whether a pricing rule applies to a given ride. For this to be the case, the stored locations must be comparable to the location of the passenger, or the destination. The user must be able to reason about his pricing rules, from which an understanding of his defined locations logically follows. But edge cases must be covered completely. For example, a rule in the current system dictates that a user traveling to Schiphol should receive a discount. But how would the system detect that this is the case? Or what if hotel guests receive discounts, but the neighbour shouldn't be allowed to use these discounts? Secondly, a system has to be developed that encapsulates the solution that is the result of the conducted research. It is helpful to extend the research of the problem to finding out how to incorporate the answers into a working system, where architecture has a major influence in the tools that are available. For example: if a solution to the main problem requires a database system capable of handling high quantities of geospatial queries, this requirement has to be satisfied in order to proceed in finding the final solution. Finally, a user interface has to be created that enables users to define the pricing rules. The complexity of the interface depends on how straight forward the price calculation system is put together. The user interface should also be available in multiple portals. The best way of making the systems capabilities available to the user through the UI in the portal, must be investigated. The UI must be built keeping the user in mind, simplifying complex rule management as much as possible.

1.3.1 Questions

From the description of the problem, one main important research question can be derived: *How can a generic location-based price calculation system be implemented that is usable around the globe?*

This question encapsulates the three important challenges that have to be dealt with before the project can successfully be implemented. In order to give a clear direction to the research, sub-questions are separated into three groups; location mapping, architecture and user interface.

1. In what way can locations be represented to be universally interpretable?

- 1.1. Which types of locations should be distinguished?
- 1.2. What are the main differences between postal systems used around the globe?
- 1.3. Can postal codes be abstracted to geospatial data while retaining the same usefulness in the system?
- 1.4. How can different types of locations be effectively stored in a database?
2. Which architectural pattern is best suited for implementing the pricing system?
 - 2.1. Which architectural patterns fit in with the existing architecture?
 - 2.2. How will authentication be handled?
3. How can the system be implemented while so that defining rules be as insightful as possible to the user?
 - 3.1. Which Database Management System (DBMS) is suited for this project?
 - 3.2. Which views should exist, does a logical hierarchy exist among views?
 - 3.3. How should locations be defined and managed by the user?
 - 3.4. How should timeframes be handled in the interface?

The first group of questions is answered in the chapter

kks32: Add chapter name

. The second and third is answered in the chapter Proposed Approach. At that point, enough knowledge is available to implement a solution.

1.4 Process

Scrum is used to iteratively implement the solution. This is taxiID's first project to make use of Scrum and Jira for project management as a pilot. Therefore, some methods may differ from conventional prescribed methods. Before the start of the project, a pregame document was written to capture the project vision, the interpretation and proposed solution in that order and contains research information from this thesis. This, to make clear whether the requirements were correctly interpreted, and the proposed architectural vision aligns with, and correctly incorporates these requirements. The pregame document is added as an appendix

kks32: Add pregame to appendix

, and provides more insight in the requirement gathering before research was conducted.

Chapter

kks32: Refer to chapter containing methods and techniques about scrum

.

Chapter 2

Encoding Locations

2.1 Introduction

To answer in what way locations can be represented to be universally interpretable, the definition of a location must be well understood. Encoding of locations has historically been of great importance, and is always being modernized. This chapter aims to find the best method of representing locations that is universally interpretable and most importantly usable in this project. GPS and Postal Codes are globally adopted and have proven to be reliable methods of encoding locations. Locations, plurally, suggesting that more than one location should be encoded. This is where the challenge lies.

2.2 A Brief History Of Geographic Locations

A location is roughly described as a place or position. Throughout history, various navigational techniques and tools like the sextant, nautical chart and mariner's compass were used, measuring the altitude of the North Star to determine the latitude ϕ , in conjunction with a chronometer to determine the longitude λ of a location on the Earth's surface.

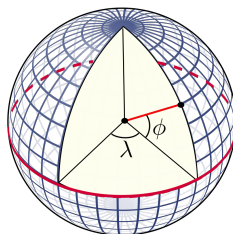


Fig. 2.1 A perspective view of the Earth showing how latitude and longitude are defined on a spherical model.

Modern navigation relies on satellites that are capable of providing information to determine a location with an accuracy of 9 meters. The precision of hybrid methods using cell towers and Wi-Fi Location Services enables precise tracking of modern devices. Latitude and longitude are often referred to as GPS coordinates, as GPS is often used to calculate latitude and longitude by receiving position and time code sequences from at least four satellites. Addresses are another representation of a location used in navigation. Addresses are easier to communicate than a pair of GPS coordinates, but can be ambiguous, imprecise, inconsistent in format. Addresses commonly make use of Postal Code systems, which have reliably been assigned to geographical areas with the purpose of sorting mail. Although even today, there are countries that do not have a Postal Code system. A location being roughly described as a place or position, can be decomposed as an abstract term to describe physical or imaginary areas with varying radiusses and shapes. You could prepend 'the location of' to the following terms as an example: America, the birthplace of Sokrates, Wall Street, the center of the universe, the Laryngeal Nerve of the Giraffe, churches in the Netherlands. The final example presents the main challenge of this project.

2.3 Useful Location Types

While setting up a backlog, a shared knowledge about the terminology used in the issues must be achieved. In the pregame document, the term "area" was defined as a collection of three or more coordinate pairs, or a collection of postal codes. A "point" was defined as one distinct coordinate pair or one distinct postal code. This way a location could either be an area or a point, with which all possibilities are covered. As stated in appendix

kks32: appendix pregame ref

definition of an area is precise, unambiguous and easy to use in compare in computer programs. A single point may match another single point if it's the exact same point. A point may be sitting on top of a line or is contained within an area. The only other option is the negation of these statements. Because use cases for lines will be non-existent, points and areas are the proper candidates for spatial queries.

2.4 Requisites of Locations

A taxi company director wants to be able to set price or define discounts from or to a certain location. They would like to define prices based only on departure locations, or only on destination locations, or both. For example: 'to Schiphol, a trip should cost €10,-', or 'from

Falke hotels a trip should cost €5,-', or 'from Falke hotels to Schiphol, the km price should be €0,60'. In the current implementation, a record would be stored containing departure location, destination location and price for every combination, where locations were defined as zip codes. Instead, it would make sense to be able to reuse locations after they have been defined once.

2.5 Literature Review

In what way can locations be represented to be universally interpretable?

1. Which types of locations should be distinguished?
2. What are the main differences between postal systems used around the globe?
3. Can postal codes be abstracted to geospatial data while retaining the same usefulness in the system?
4. How can different types of locations be effectively stored in a database?

Chapter 3

System Architecture

3.1 Introduction

The price calculation system that is being made should integrate in the existing architecture seamlessly. This chapter aims to answer question number three, and its subquestions. First it is determined whether the frontend and backend that are to be developed should be integrated in existing projects, or should be made in separate projects altogether. If the backend is created as a separate project, authentication and authorization are directly affected by these decisions. Separation implies more complex identity management, or less separation of concern. Then, the database should be capable of storing geometry, and accept queries to determine which polygons contain a set of points, and which points are contained within a polygon.

3.2 Architectural Patterns

The current system architecture consists of three public API's and eight private API's that connect to four databases.

kks32: Refer to figure

The core API is the interface to which mobile applications make requests. It is possible to integrate the pricing system as a module in the existing project. This simplifies authentication and authorization, because it already exists in that project. Another option is to implement the system as a microservice, which is infamously known as a service-oriented architecture (SOA). Advantages of a microservice are the fact that a microservice is a self-contained and naturally modular structure, but authentication and authorization must be handled by the microservice itself, unless state is shared amongst services.

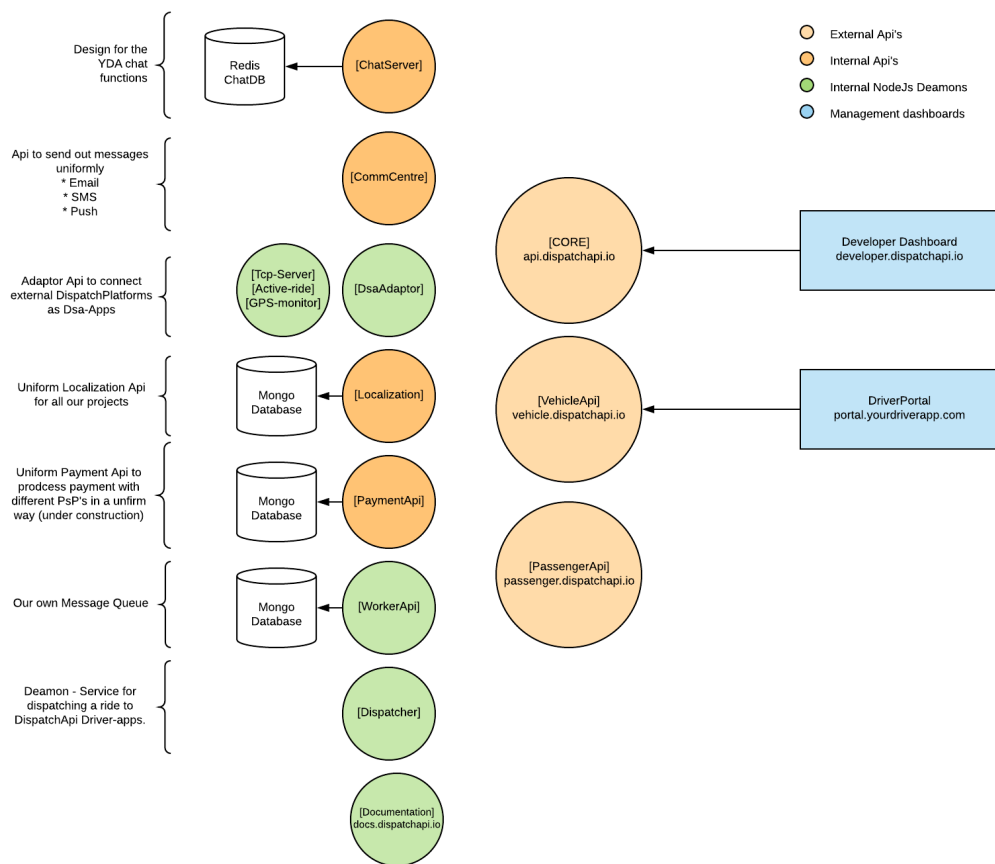


Fig. 3.1 Current System Architecture

3.3 Authentication and Authorization

Mobile applications should be able to make requests, just like the portals that are to be developed. But portal users make use of the microservice in a different way. Mobile apps merely request prices of products, based on the rules that group admins define through the portal. To make sure that only the portal users have the right to mutate their data, users have to be authenticated and authorized within the microservice. Identity management becomes a problem if data duplication is not desired. If a user makes a direct request to the microservice, the credentials have to be compared to user data in a database. To prevent duplication, the microservice could be connected to the database that is used by the core system. But this makes the microservice less decoupled, and directly contradicts the desire to separate concerns. Four examples demonstrate this problem:

- Example 1: The microservice authenticates and authorizes users all by itself, managing sessions and storing user data in its database.
- Example 2: The microservice connects to an existing databases to acquire the required information about the user.
- Example 3: The core system authenticates the user and provides a token that can be verified by the microservice, containing user identity.
- Example 4: A separate service is used for authentication and authorization so that the core system is not involved at all.

In the first example, the microservice seems to be independent, because it has knowledge about the users identity without making requests to adjacent systems, or connecting to external databases. But this is not true. If data about the user is mutated in the core system, the microservice needs to be notified or synced. This greatly hinders scaling and makes it harder to keep data consistent. Example two solves the inconsistency part by connecting to the central database that holds user data, but contradicts the strive for encapsulation. Example three entirely removes the database connection to any user data. This is possible when a JSON Web Token (JWT) is used for example. The identity is stored in the token itself as an encrypted payload that can only be revealed by whoever holds the secret with which it was signed. Example four delegates managing user identity to a separate authentication service that, similar to the pricing microservice, has its own single task.

Which architectural pattern is best suited for implementing the pricing system?

1. Which architectural patterns fit in with the existing architecture?
2. How will authentication be handled?

Chapter 4

Proposed Solution

4.1 Introduction

4.2 Design

How can the task of defining rules be as insightful as possible to the user?

1. Which database is suited for this project?
2. Which views should exist, does a logical hierarchy exist among views?
3. How should locations be defined and managed by the user?
4. How should timeframes be handled in the interface?

4.3 Methods and Techniques

Chapter 5

Realization

5.1 What is lorem ipsum? Title with math σ

Lorem Ipsum is simply dummy text of the printing and typesetting industry (see Section 5.3). Lorem Ipsum [2] has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum [1, 3, 4].

The most famous equation in the world: $E^2 = (m_0c^2)^2 + (pc)^2$, which is known as the **energy-mass-momentum** relation as an in-line equation.

A *LaTeX class file* is a file, which holds style information for a particular *LaTeX*.

$$CIF : \quad F_0^j(a) = \frac{1}{2\pi i} \oint_{\gamma} \frac{F_0^j(z)}{z-a} dz \quad (5.1)$$

5.2 Why do we use lorem ipsum?

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many

web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

5.3 Where does it come from?

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, *consectetur*, from a Lorem Ipsum passage, and going through the cites of the word in classical literature, discovered the undoubtable source. Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum, "Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham

"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

Section 1.10.32 of "de Finibus Bonorum et Malorum", written by Cicero in 45 BC: "Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?"

1914 translation by H. Rackham: "But I must explain to you how all this mistaken idea of denouncing pleasure and praising pain was born and I will give you a complete

account of the system, and expound the actual teachings of the great explorer of the truth, the master-builder of human happiness. No one rejects, dislikes, or avoids pleasure itself, because it is pleasure, but because those who do not know how to pursue pleasure rationally encounter consequences that are extremely painful. Nor again is there anyone who loves or pursues or desires to obtain pain of itself, because it is pain, but because occasionally circumstances occur in which toil and pain can procure him some great pleasure. To take a trivial example, which of us ever undertakes laborious physical exercise, except to obtain some advantage from it? But who has any right to find fault with a man who chooses to enjoy a pleasure that has no annoying consequences, or one who avoids a pain that produces no resultant pleasure?"

Section 1.10.33 of "de Finibus Bonorum et Malorum", written by Cicero in 45 BC: "At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident, similique sunt in culpa qui officia deserunt mollitia animi, id est laborum et dolorum fuga. Et harum quidem rerum facilis est et expedita distinctio. Nam libero tempore, cum soluta nobis est eligendi optio cumque nihil impedit quo minus id quod maxime placeat facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores alias consequatur aut perferendis doloribus asperiores repellat."

1914 translation by H. Rackham: "On the other hand, we denounce with righteous indignation and dislike men who are so beguiled and demoralized by the charms of pleasure of the moment, so blinded by desire, that they cannot foresee the pain and trouble that are bound to ensue; and equal blame belongs to those who fail in their duty through weakness of will, which is the same as saying through shrinking from toil and pain. These cases are perfectly simple and easy to distinguish. In a free hour, when our power of choice is untrammelled and when nothing prevents our being able to do what we like best, every pleasure is to be welcomed and every pain avoided. But in certain circumstances and owing to the claims of duty or the obligations of business it will frequently occur that pleasures have to be repudiated and annoyances accepted. The wise man therefore always holds in these matters to this principle of selection: he rejects pleasures to secure other greater pleasures, or else he endures pains to avoid worse pains."

Chapter 6

Conclusion

6.1 Reasonably long section title

I'm going to randomly include a picture Figure 6.1.

If you have trouble viewing this document contact Krishna at: kks32@cam.ac.uk or raise an issue at <https://github.com/kks32/phd-thesis-template/>

Enumeration

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed vitae laoreet lectus. Donec lacus quam, malesuada ut erat vel, consectetur eleifend tellus. Aliquam non feugiat lacus. Interdum et malesuada fames ac ante ipsum primis in faucibus. Quisque a dolor sit amet dui malesuada malesuada id ac metus. Phasellus posuere egestas mauris, sed porta arcu vulputate ut. Donec arcu erat, ultrices et nisl ut, ultricies facilisis urna. Quisque iaculis, lorem non maximus pretium, dui eros auctor quam, sed sodales libero felis vel orci. Aliquam neque nunc, elementum id accumsan eu, varius eu enim. Aliquam blandit ante et ligula tempor pharetra. Donec molestie porttitor commodo. Integer rutrum turpis ac erat tristique cursus. Sed venenatis urna vel tempus venenatis. Nam eu rhoncus eros, et condimentum elit. Quisque risus turpis, aliquam eget euismod id, gravida in odio. Nunc elementum nibh risus, ut faucibus mauris molestie eu. Vivamus quis nunc nec nisl vulputate fringilla. Duis tempus libero ac justo laoreet tincidunt. Fusce sagittis gravida magna, pharetra venenatis mauris semper at. Nullam eleifend felis a elementum sagittis. In vel turpis eu metus euismod tempus eget sit amet tortor. Donec eu rhoncus libero, quis iaculis lectus. Aliquam erat volutpat. Proin id ullamcorper tortor. Fusce vestibulum a enim non volutpat. Nam ut interdum nulla. Proin lacinia felis malesuada arcu aliquet fringilla. Aliquam condimentum, tellus eget maximus

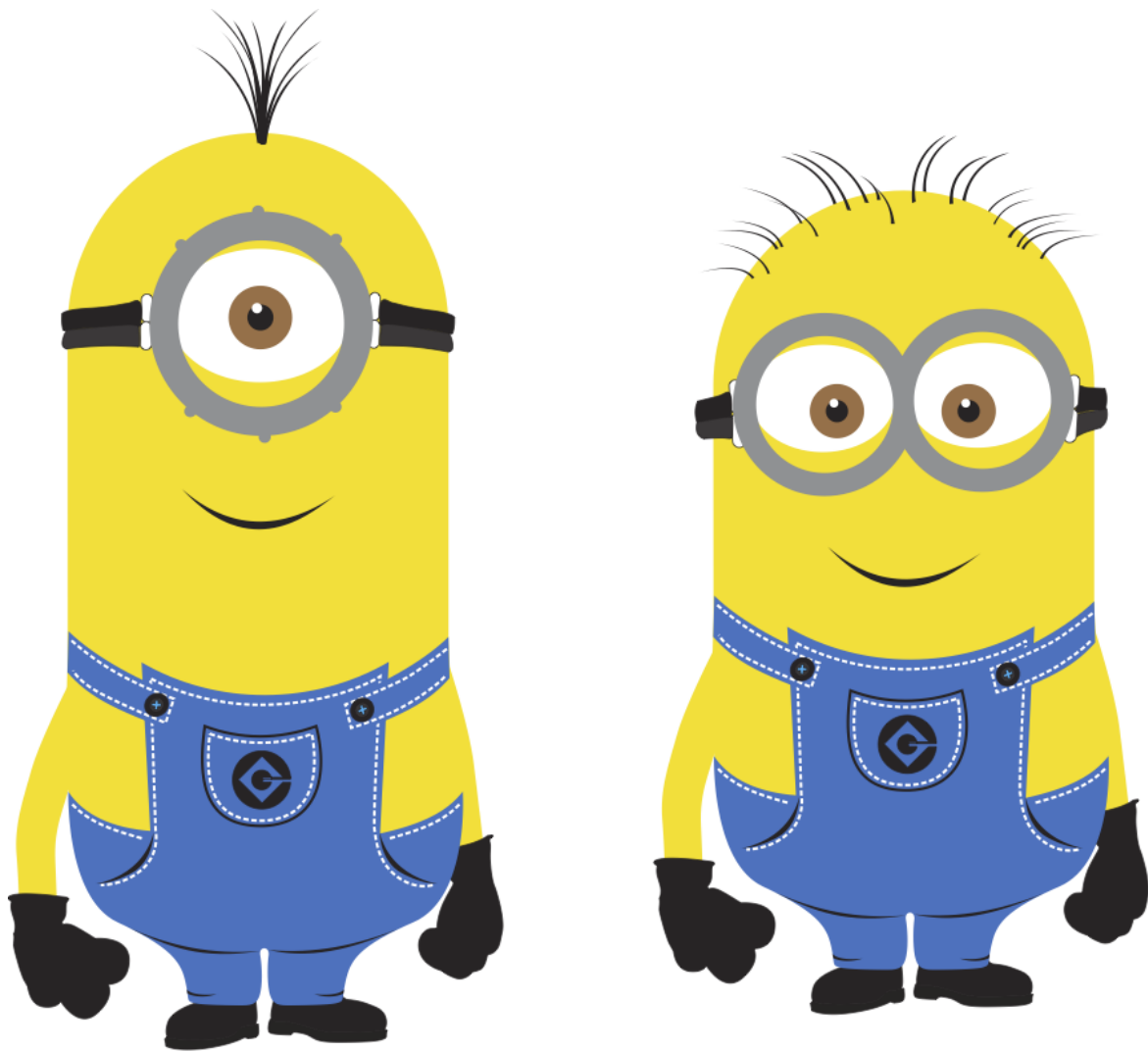


Fig. 6.1 This is just a long figure caption for the minion in Despicable Me from Pixar

porttitor, quam sem luctus massa, eu fermentum arcu diam ac massa. Praesent ut quam id leo molestie rhoncus. Praesent nec odio eget turpis bibendum eleifend non sit amet mi. Curabitur placerat finibus velit, eu ultricies risus imperdiet ut. Suspendisse lorem orci, luctus porta eros a, commodo maximus nisi.

Nunc et dolor diam. Phasellus eu justo vitae diam vehicula tristique. Vestibulum vulputate cursus turpis nec commodo. Etiam elementum sit amet erat et pellentesque. In eu augue sed tortor mollis tincidunt. Mauris eros dui, sagittis vestibulum vestibulum vitae, molestie a velit. Donec non felis ut velit aliquam convallis sit amet sit amet velit. Aliquam vulputate, elit in lacinia lacinia, odio lacus consectetur quam, sit amet facilisis mi justo id magna. Curabitur aliquet pulvinar eros. Cras metus enim, tristique ut magna a, interdum egestas nibh. Aenean lorem odio, varius a sollicitudin non, cursus a odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae;

1. The first topic is dull
2. The second topic is duller
 - (a) The first subtopic is silly
 - (b) The second subtopic is stupid
3. The third topic is the dullest

Morbi bibendum est aliquam, hendrerit dolor ac, pretium sem. Nunc molestie, dui in euismod finibus, nunc enim viverra enim, eu mattis mi metus id libero. Cras sed accumsan justo, ut volutpat ipsum. Nam faucibus auctor molestie. Morbi sit amet eros a justo pretium aliquet. Maecenas tempor risus sit amet tincidunt tincidunt. Curabitur dapibus gravida gravida. Vivamus porta ullamcorper nisi eu molestie. Ut pretium nisl eu facilisis tempor. Nulla rutrum tincidunt justo, id placerat lacus laoreet et. Sed cursus lobortis vehicula. Donec sed tortor et est cursus pellentesque sit amet sed velit. Proin efficitur posuere felis, porta auctor nunc. Etiam non porta risus. Pellentesque lacinia eros at ante iaculis, sed aliquet ipsum volutpat. Suspendisse potenti.

Ut ultrices lectus sed sagittis varius. Nulla facilisi. Nullam tortor sem, placerat nec condimentum eu, tristique eget ex. Nullam pretium tellus ut nibh accumsan elementum. Aliquam posuere gravida tellus, id imperdiet nulla rutrum imperdiet. Nulla pretium ullamcorper quam, non iaculis orci consectetur eget. Curabitur non laoreet nisl. Maecenas lacinia, lorem vel tincidunt cursus, odio lorem aliquet est, gravida auctor arcu urna id enim. Morbi accumsan bibendum ipsum, ut maximus dui placerat vitae. Nullam pretium ac tortor nec venenatis. Nunc non aliquet neque.

Itemize

- The first topic is dull
- The second topic is duller
 - The first subtopic is silly
 - The second subtopic is stupid
- The third topic is the dullest

Description

The first topic is dull

The second topic is duller

The first subtopic is silly

The second subtopic is stupid

The third topic is the dullest

6.2 Hidden section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In magna nisi, aliquam id blandit id, congue ac est. Fusce porta consequat leo. Proin feugiat at felis vel consectetur. Ut tempus ipsum sit amet congue posuere. Nulla varius rutrum quam. Donec sed purus luctus, faucibus velit id, ultrices sapien. Cras diam purus, tincidunt eget tristique ut, egestas quis nulla. Curabitur vel iaculis lectus. Nunc nulla urna, ultrices et eleifend in, accumsan ut erat. In ut ante leo. Aenean a lacinia nisl, sit amet ullamcorper dolor. Maecenas blandit, tortor ut scelerisque congue, velit diam volutpat metus, sed vestibulum eros justo ut nulla. Etiam nec ipsum non enim luctus porta in in massa. Cras arcu urna, malesuada ut tellus ut, pellentesque mollis risus. Morbi vel tortor imperdiet arcu auctor mattis sit amet eu nisi. Nulla gravida urna vel nisl egestas varius. Aliquam posuere ante quis malesuada dignissim. Mauris ultrices tristique eros, a dignissim nisl iaculis nec. Praesent dapibus tincidunt mauris nec tempor. Curabitur et consequat nisi. Quisque viverra egestas risus, ut sodales enim blandit at. Mauris quis odio nulla. Cras euismod turpis magna, in facilisis diam congue non. Mauris faucibus nisl a orci dictum, et tempus mi cursus.

Etiam elementum tristique lacus, sit amet eleifend nibh eleifend sed ¹. Maecenas dapibus augue ut urna malesuada, non tempor nibh mollis. Donec sed sem sollicitudin, convallis velit aliquam, tincidunt diam. In eu venenatis lorem. Aliquam non augue porttitor tellus faucibus porta et nec ante. Proin sodales, libero vitae commodo sodales, dolor nisi cursus magna, non tincidunt ipsum nibh eget purus. Nam rutrum tincidunt arcu, tincidunt vulputate mi sagittis id. Proin et nisi nec orci tincidunt auctor et porta elit. Praesent eu dolor ac magna cursus euismod. Integer non dictum nunc.

¹My footnote goes blah blah blah! ...



Fig. 6.2 Best Animations

Subplots

I can cite Wall-E (see Fig. 6.2b) and Minions in despicable me (Fig. 6.2c) or I can cite the whole figure as Fig. 6.2

References

- [1] Abramovich, Y. A., Aliprantis, C. D., and Burkinshaw, O. (1995). Another characterization of the invariant subspace problem. *Operator Theory in Function Spaces and Banach Lattices*. The A.C. Zaanen Anniversary Volume, *Operator Theory: Advances and Applications*, 75:15–31. Birkhäuser Verlag.
- [2] Aupetit, B. (1991). *A Primer on Spectral Theory*. Springer-Verlag, New York.
- [3] Conway, J. B. (1990). *A Course in Functional Analysis*. Springer-Verlag, New York, second edition.
- [4] Ljubič, J. I. and Macaev, V. I. (1965). On operators with a separable spectrum. *Amer. Math. Soc. Transl. (2)*, 47:89–129.

Appendix A

How to install L^AT_EX

Windows OS

TeXLive package - full version

1. Download the TeXLive ISO (2.2GB) from
<https://www.tug.org/texlive/>
2. Download WinCDEmu (if you don't have a virtual drive) from
<http://wincdemu.sysprogs.org/download/>
3. To install Windows CD Emulator follow the instructions at
<http://wincdemu.sysprogs.org/tutorials/install/>
4. Right click the iso and mount it using the WinCDEmu as shown in
<http://wincdemu.sysprogs.org/tutorials/mount/>
5. Open your virtual drive and run setup.pl

or

Basic MikTeX - T_EX distribution

1. Download Basic-MiK_TE_X(32bit or 64bit) from
<http://miktex.org/download>
2. Run the installer
3. To add a new package go to Start » All Programs » MikTeX » Maintenance (Admin)
and choose Package Manager

4. Select or search for packages to install

TexStudio - T_EX editor

1. Download TexStudio from
<http://texstudio.sourceforge.net/#downloads>
2. Run the installer

Mac OS X

MacTeX - T_EX distribution

1. Download the file from
<https://www.tug.org/mactex/>
2. Extract and double click to run the installer. It does the entire configuration, sit back and relax.

TexStudio - T_EX editor

1. Download TexStudio from
<http://texstudio.sourceforge.net/#downloads>
2. Extract and Start

Unix/Linux

TeXLive - T_EX distribution

Getting the distribution:

1. TeXLive can be downloaded from
<http://www.tug.org/texlive/acquire-netinstall.html>.
2. TeXLive is provided by most operating system you can use (rpm,apt-get or yum) to get TeXLive distributions

Installation

1. Mount the ISO file in the mnt directory

```
mount -t iso9660 -o ro,loop,noauto /your/texlive####.iso /mnt
```

2. Install wget on your OS (use rpm, apt-get or yum install)
3. Run the installer script install-tl.

```
cd /your/download/directory
./install-tl
```

4. Enter command 'i' for installation
5. Post-Installation configuration:
<http://www.tug.org/texlive/doc/texlive-en/texlive-en.html#x1-320003.4.1>
6. Set the path for the directory of TexLive binaries in your .bashrc file

For 32bit OS

For Bourne-compatible shells such as bash, and using Intel x86 GNU/Linux and a default directory setup as an example, the file to edit might be

```
edit ~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/i386-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;
export INFOPATH
```

For 64bit OS

```
edit ~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/x86_64-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
```

```
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;  
export INFOPATH
```

Fedora/RedHat/CentOS:

```
sudo yum install texlive  
sudo yum install psutils
```

SUSE:

```
sudo zypper install texlive
```

Debian/Ubuntu:

```
sudo apt-get install texlive texlive-latex-extra  
sudo apt-get install psutils
```

Appendix B

Installing the CUED class file

\LaTeX .cls files can be accessed system-wide when they are placed in the $\langle\text{texmf}\rangle/\text{tex}/\text{latex}$ directory, where $\langle\text{texmf}\rangle$ is the root directory of the user's \TeX installation. On systems that have a local texmf tree ($\langle\text{texmflocal}\rangle$), which may be named “ texmf-local ” or “ localtexmf ”, it may be advisable to install packages in $\langle\text{texmflocal}\rangle$, rather than $\langle\text{texmf}\rangle$ as the contents of the former, unlike that of the latter, are preserved after the \LaTeX system is reinstalled and/or upgraded.

It is recommended that the user create a subdirectory $\langle\text{texmf}\rangle/\text{tex}/\text{latex}/\text{CUED}$ for all CUED related \LaTeX class and package files. On some \LaTeX systems, the directory look-up tables will need to be refreshed after making additions or deletions to the system files. For \TeX Live systems this is accomplished via executing “ texhash ” as root. MikTeX users can run “ initexmf -u ” to accomplish the same thing.

Users not willing or able to install the files system-wide can install them in their personal directories, but will then have to provide the path (full or relative) in addition to the filename when referring to them in \LaTeX .

Index

LaTeX class file, 17