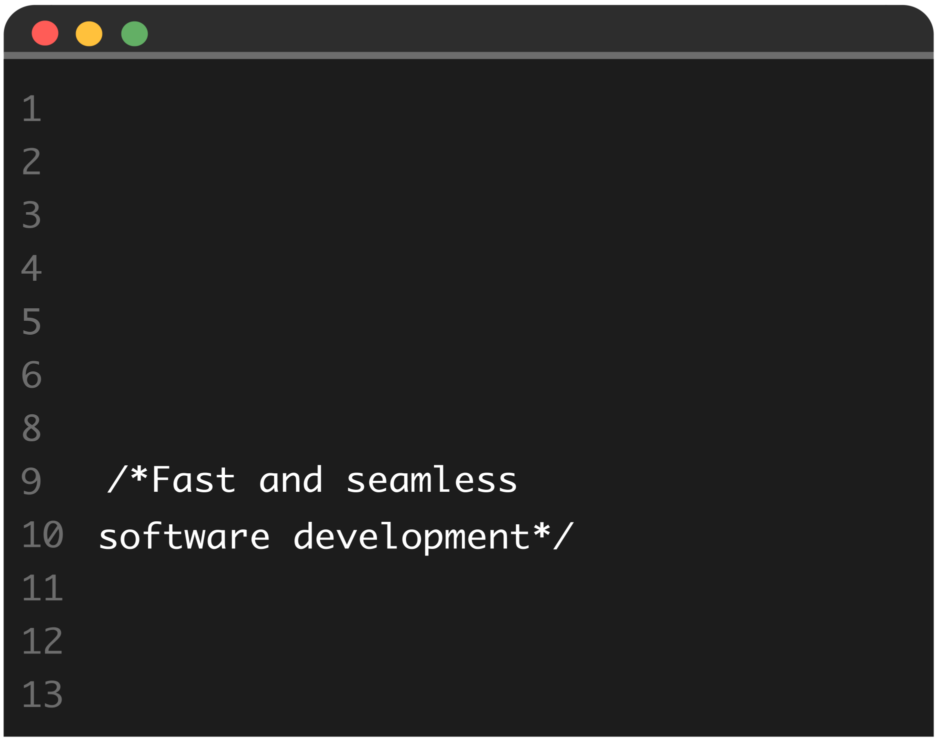
# An IDE called Vim



```
 /*Fast and seamless
software development*/
```

Cláudio Ribeiro

# An IDE Called Vim

Claudio Ribeiro

This book is for sale at http://leanpub.com/anidecalledvim

This version was published on 2022-01-29

# Contents

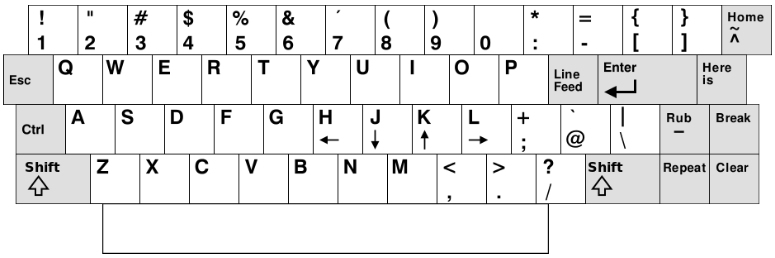
# Move Around

In the previous section, we talked about Vim modes and introduced some very basic commands and motions. We also saw how we could change from mode to mode. In this chapter, we will cover basic movement on Vim in a more structured and detailed way.

When it comes to moving around, Vim is also pretty unique.

As we know, Vim is an evolution of Vi, the text editor from 1976.

In 1976, Bill Joy(2), the creator of Vi used an ADM-3A(3) terminal. What was particular about the ADM-3A is that its keyboard was a bit different from what we are used to today. It didn't have the cursor keys, and the `<ESC>` key was located where we have our `<CAPS LOCK>` key now.



**ADM-3A keyboard**

This little difference influences tremendously the way we use Vim. Though it is possible to use the cursor keys to move around in Vim, we normally opt to use the keys used in the original Vi version: `h`, `j`, `k`, `l`, for `left`, `down`, `up` and `right` respectively. Why do we do this? Because it allows us to rest our hands on the home row and not waste movement between the cursor keys and the rest of the keyboard.

The fact that the `<ESC>` also used to be in the home row reinforces the fact that no movement was wasted to change between modes while using Vi.

This “**No Wasted Movement**” philosophy is what makes Vim a very powerful and productive text editor, and from now on I recommend getting used to the `h`, `j`, `k` and `l` keys to move around.

## Navigating from mode to mode

When we’re talking about moving around Vim, being able to move from mode to mode is a top priority.

Even though we covered Vim modes and how to reach them in a past chapter, I think we should get a second look at them right now.

Normal mode is where we will be for most of our time. So our focus will be reaching all the other modes through Normal mode. Remember that at any time we can switch to Normal mode by hitting the `<ESC>` key.

To reach Visual mode we hit the `<v>` key from Normal mode.

To reach Command line mode we hit `<:>` from Normal mode.

Insert mode is the one that has some nuances to it. So we’re going to condense all the ways we can reach it from Normal mode in the following table:

| **Insert Mode mode** | **Command** |
|---|---|
| Append after cursor | `<a>` |
| Append before cursor | `<i>` |
| Append at the end of the line | `<A>` |
| Append at the beginning of the line | `<I>` |
| Change at the end of the line | `<C>` |
| Substitute characters | `<s>` |

All the commands above change our mode to Insert mode but in different conditions. It is useful to know them, as it can speed up our code writing by a lot.

This is how we can access the most common modes in Vim. Be sure to experiment with them as they are an indispensable tool to have when working with Vim.

## Locating ourselves in a file

Another good thing to have in any text or code editor is the ability to know where we are in the file. Vim provides us that information with the `<C-g>` command. This command will provide not only the line we are on but also how many lines the file has and the column we're on. This information is provided in the following format:

```
/tmp/tutorUj1jxk" line 510 of 970 --52%-- col 1
```

We also have the ability to jump to either to start or the end of a file:

- `<G>` will move us to the bottom of the file,
- `<gg>` will move us to the top of the file.

Of course, these commands are available only in Normal mode.

## The `%` character

We end this `Moving around` section by analyzing the `<%>` command. This command is called the **Matching parenthesis search**.

When putting the cursor over any of the following `(`, `)`, `[`, `]`,`{` or `}` characters; pressing `<%>` will move the cursor automatically to the matching parenthesis or bracket.

This is a specially useful command when writing code.

There is a lot more that can be said about moving around in VIm, and a lot more commands available. This is just a collection of basics that can be used to quickstart us. When we look at *Basic Motions* later on this chapter we will see how we can move around much more quickly and accurately. But until then, these commands are more than enough to use and be productive with Vim.