CAMEL

*C*reativity *A*ided *M*odeling for *E*lectronic *L*ights

Creation Assistée pour Matrices Electroniques à Leds

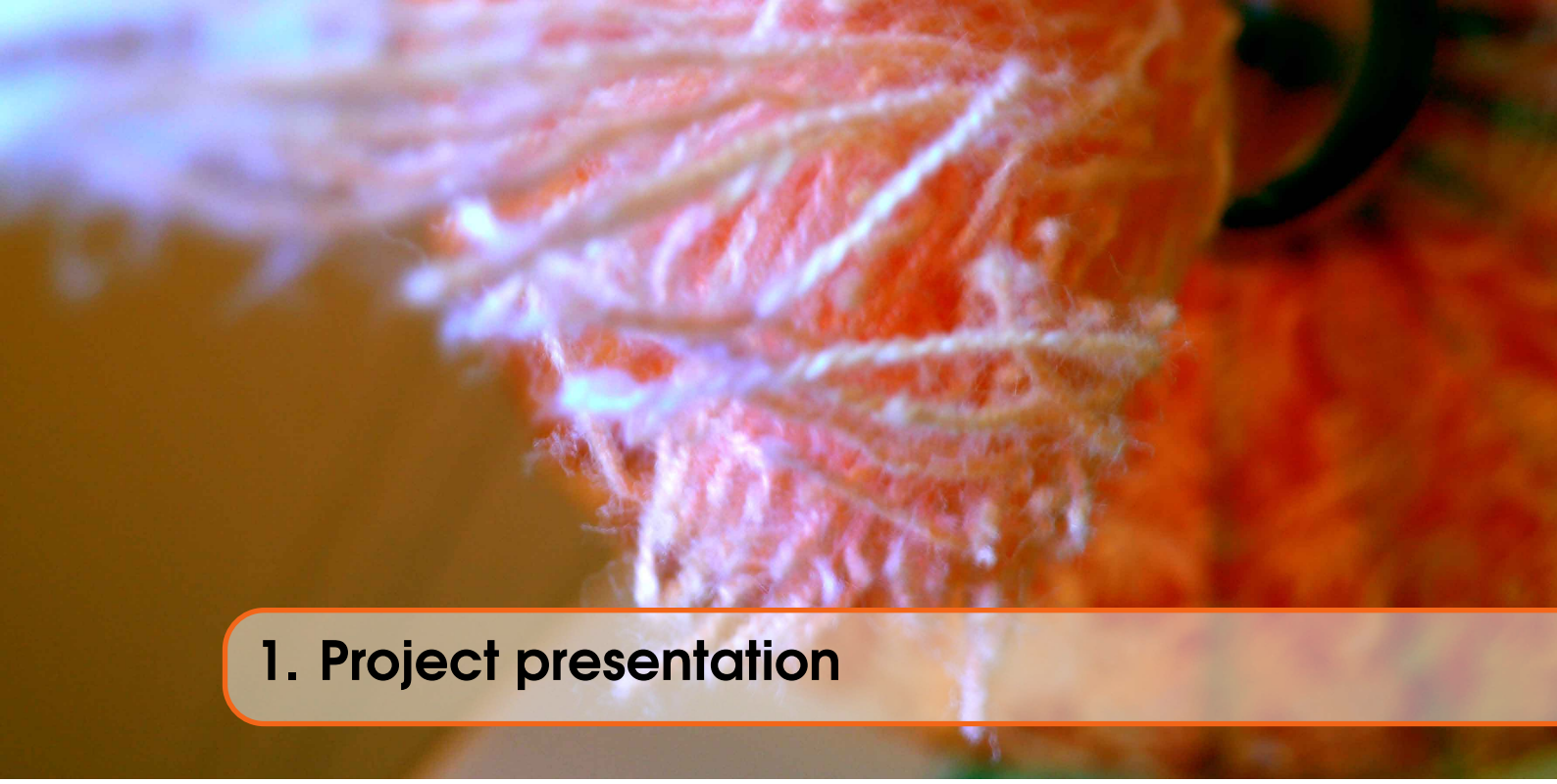
Software design document

David Lewin

First edition, Avril 2017

Contents

Part One



1. Project presentation

1.1 The origins: Where this project came from

In 2006, a project were released as the companion of the Beagle Bone Black Server book. A dedicated Github repository were created to be available for anyone who wanted to get introduced to the DIY world. Although stable and functional, the code was not intentionally 100% complete. Some parts were removed as exercises. After a while some readers have requested to improve the code of this project to be a fully functional project. But I wanted something that would bring more functionalities than the version provided with the book. Indeed, even if not so obvious, the code and the book are so related that I was obliged to "restrain" some functionalities to let most of the readers be able to try by themselves.

⚠ This is why this project is a complete new design and is no more adapted for newcomers.

1.2 The main concepts

This whole project will include a designer to create leds figures (AKA patterns), a player to animate these patterns, a client to send patterns to a distant target board connected to a matrix. The Player will have 2 versions : as client of the Sender for the network version and as pattern reader who can import saved patterns from a file.

1.2.1 Model from requirements

This software will be used in different ways. You can use it from your PC as from any board that can handle binary executable. Let's list what is needed:

1.2.2 requirement list

The previous sections get you introduced to concepts and intentions of the global project. Now let's have an exhaustive features list of what we are waiting from this project. What we want is to:

- ***Design***: a pattern or some sequences : computer
- ***Play***: a recorded pattern or some sequences : anywhere
- ***Target***: Able to play with any board that can run C++ or Python

- ***Network data format***: Able to evolve to any protocol, not only TCP. [1]
- ***Unity***: Both the designer and the player have to be in the same environment. No one stands to have different tools in in different places.
- ***Openness***: The file format exported by the designer GUI can be easily used from another code/software.
- ***Flexibility***: This is the master word in the project. For example, we want to be able to create patterns and sequences for different model of led matrices
- ***Creativity***: Like openness, creativity is required to be able to implement any matrix model from the available models.
- ***Colors***: Nowadays, there are so many displays that with different colors. The designer can handle all colors a matrix can provide.

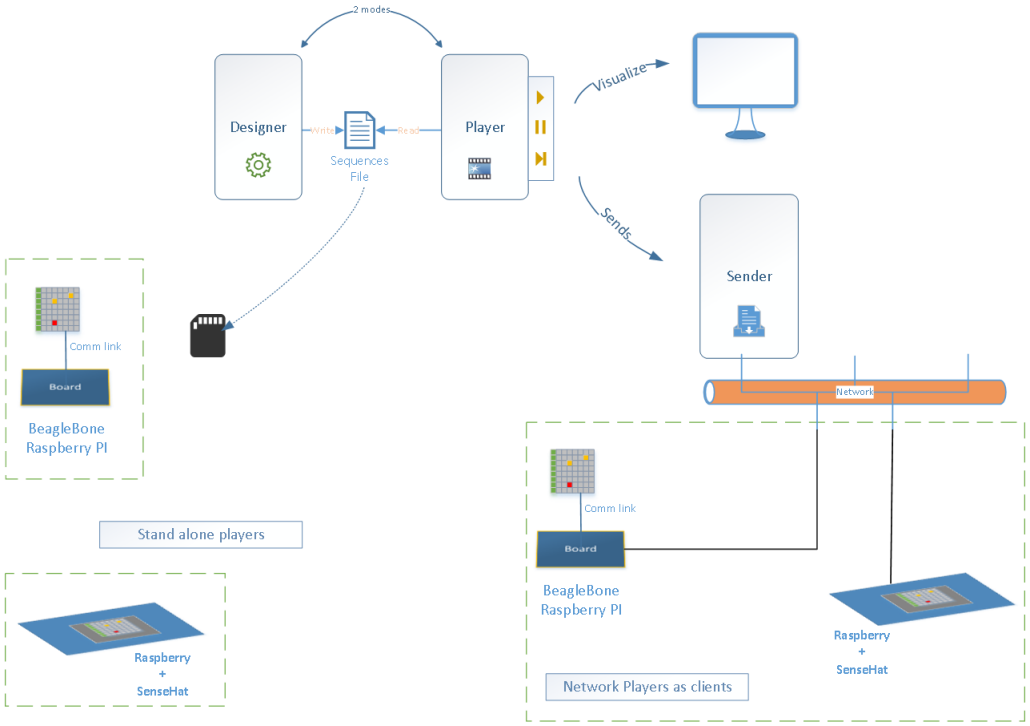


Figure 1.1: Global Architecture

1.2.3 Division into objects

After generating a set of candidate classes and objects using the classical approach and behavior analysis, the next step was to identify relationship between these objects and classes. The eligible classes and objects generated using the classical approach are shown in the Table (#ref)

Behaviors

- *User design a pattern from the design mode*
- *User design a sequence from the design mode*
- *User run a pattern from the play mode*
- *User run a sequence from the play mode*
- *Designer write a pattern from the design mode*
- *Designer write a sequence from the design mode*
- *player read a pattern from the play mode*

[1] Message queue middle ware like ZeroMQ or RabbitMQ will certainly be implemented in a further version

Actors	Events	Interactions
User	design	pattern
Player	run	sequence
Sender	Send	
Designer	Write	
Target board	open	

Table 1.1: Actors-Events-Interactions Table

- *player read a sequence from the play mode*
- *player execute a pattern/sequence from the play mode*
- *Sender send data packet*
- *Target board execute player*
- *Target board sends commands to the leds in the matrix*

Actors identification and importance

After perusing the list of possible objects associated with their usage, the Designer/Player software is the main actor in the global architecture. The definitions are in the `Design` mode when the activations and the visualization are in the `Play` mode.

The `Sender` is intended to transmit the patterns/sequences. This separated actor is able to *encapsulate* any protocol to represent the data sent to a client.

Identifying scenarii

The objects are now identified so the next step in the analysis that come in mind is : *How do these objects will interact together ?*

2 modes have been identified : DESIGN MODE and PLAY MODE. According to the previous list switching from a mode to another lets a user create patterns and test the creations directly by visualizing the animations on the computer screen. As soon as you are happy with your creations, you can send them directly with the Sender.

Later on, we will see how to do the same but with the standalone client.

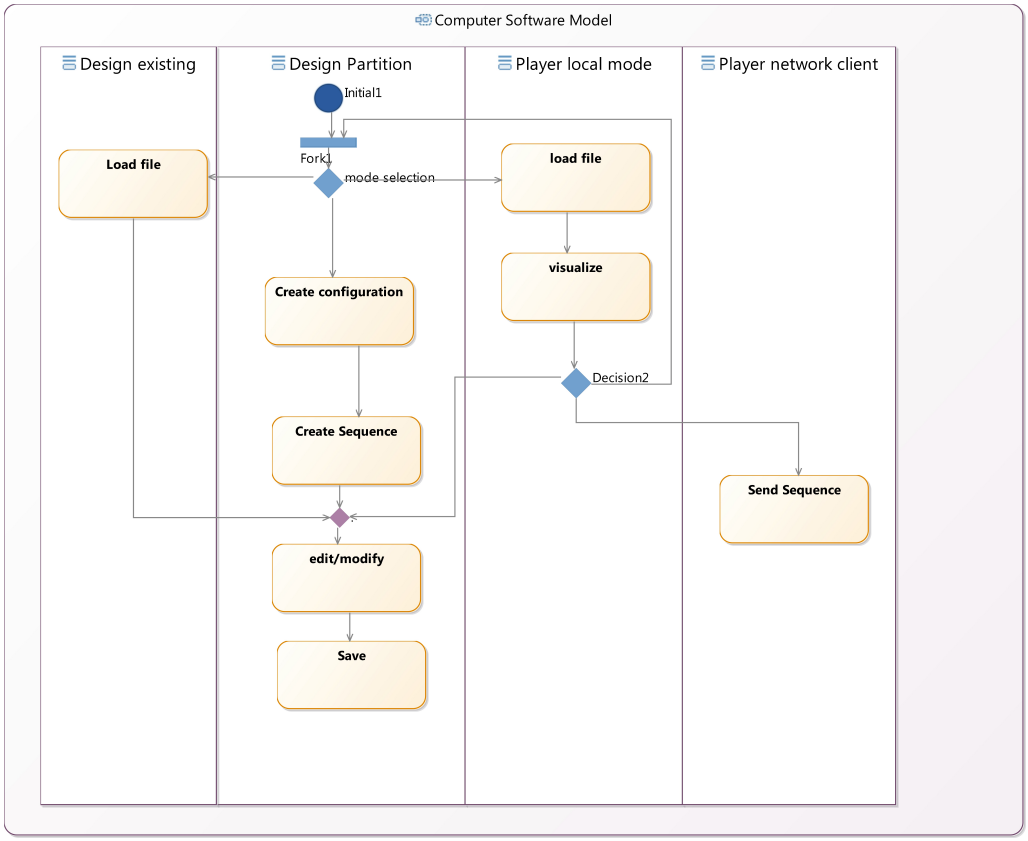


Figure 1.2: Global activity Diagram for the computer part

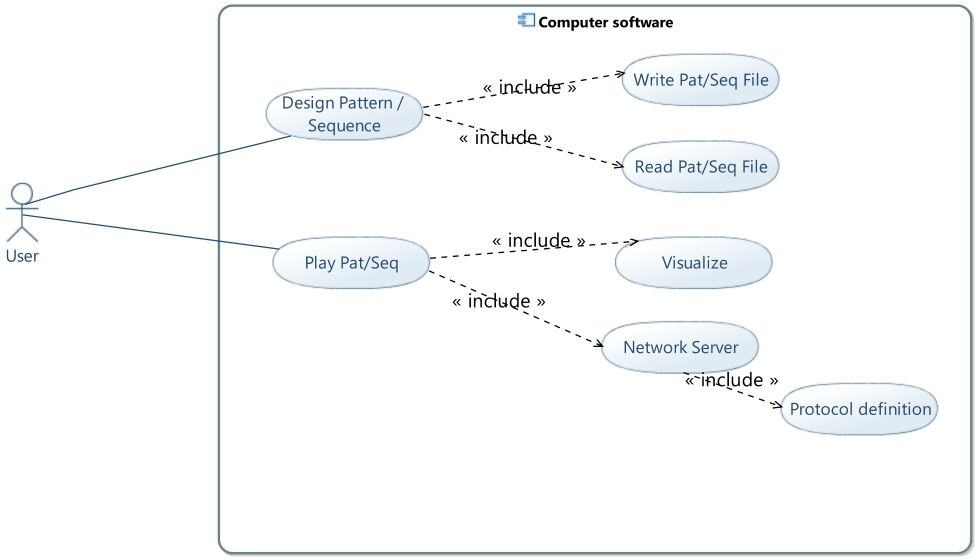


Figure 1.3: Global Computer Use Case Diagram

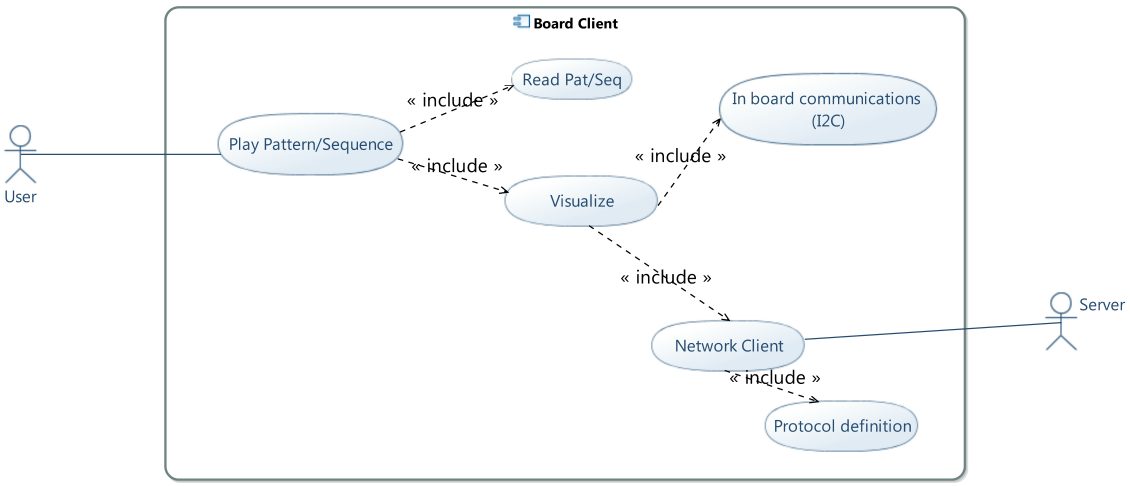


Figure 1.4: TargetBoard Global Use Case Diagram

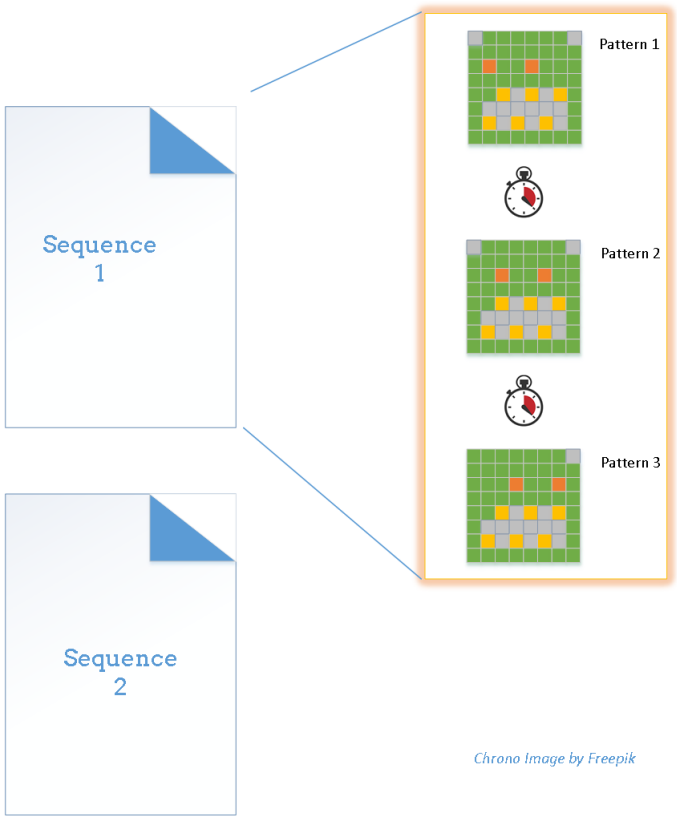


Figure 1.5: Sequence File Description

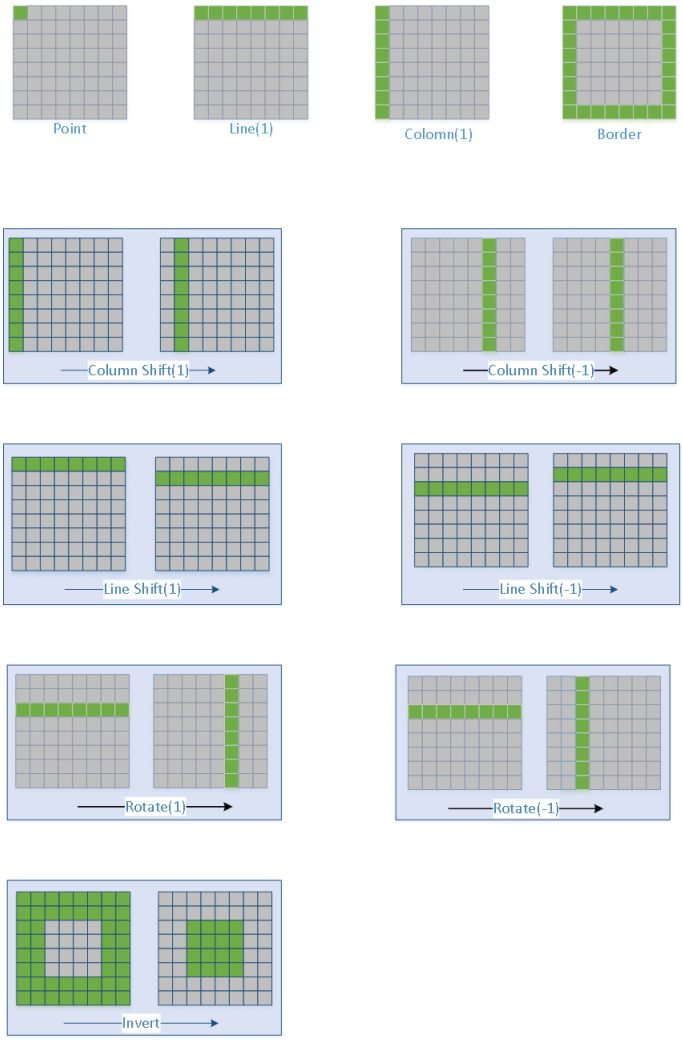


Figure 1.6: Matrix Functionalities

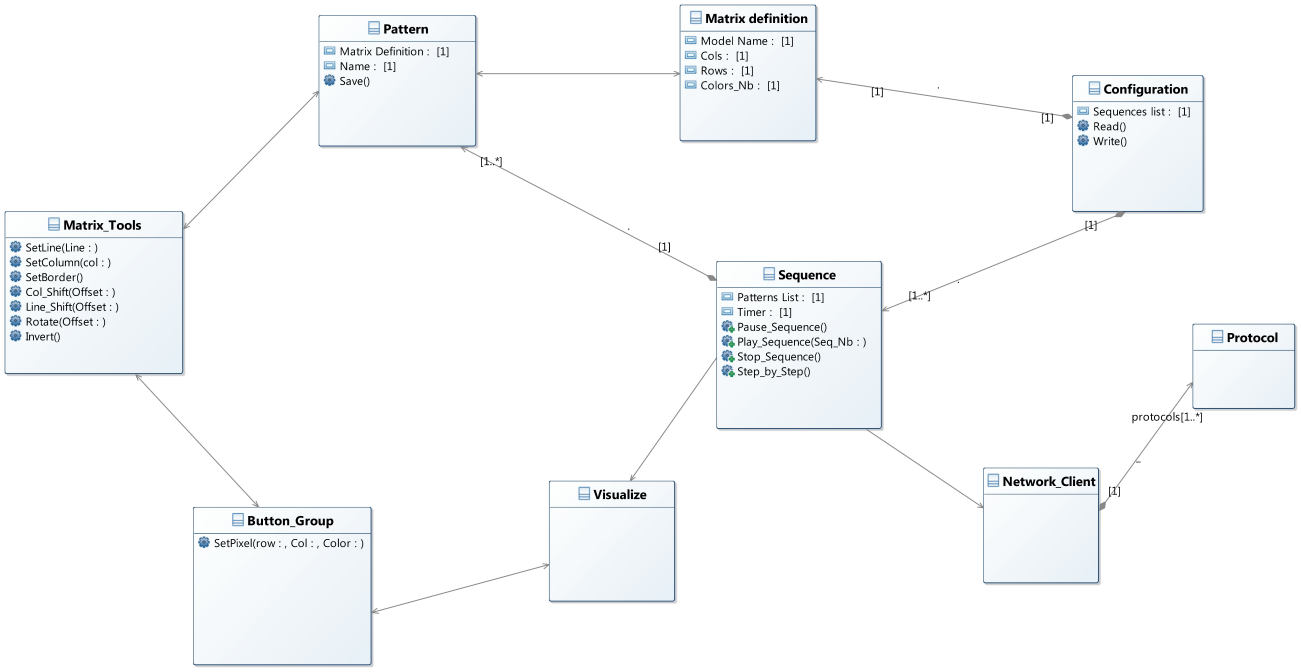


Figure 1.7: Classes Diagram Description

Defining Classes

1.2.4 Matrix used for tests

So far, the Bi color 8x8 matrix is tested. Anyway, the remaining matrices or any **square** matrix from another manufacturer can be used.

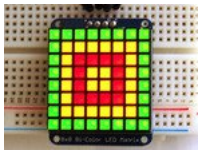


Figure 1.8: Global Bicolor pixelMatrix

R Nothing stops you to use a non square matrix like the 16x8 LED Matrix Backpack, however strange behaviors may happens at runtime with shift or rotate functions as they relies on the matrix size. However, the current code can be extended to include a special display like this one.

1.2.5 Vocabulary and definitions

Computer Windows, Linux or Apple

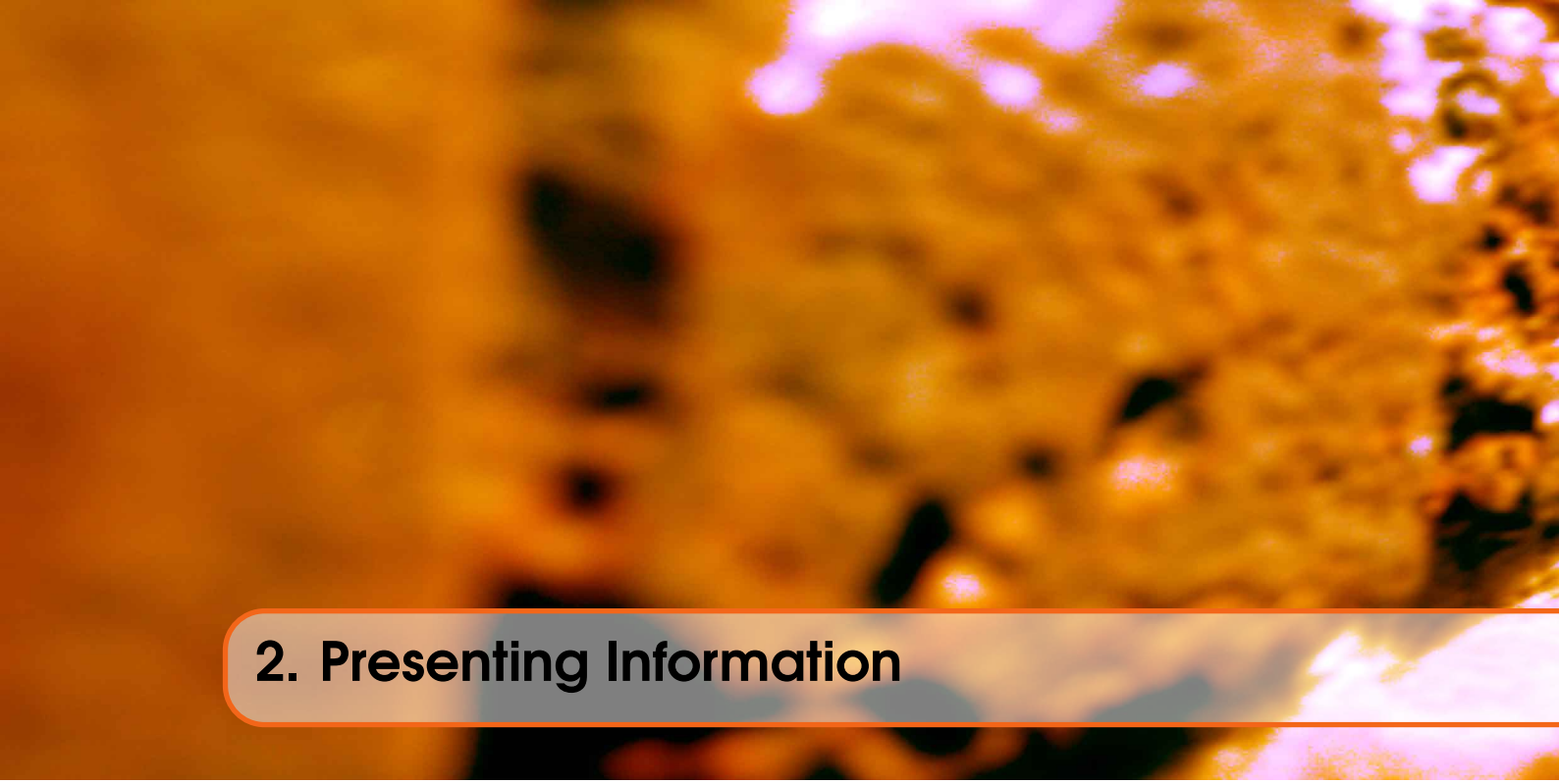
Anywhere Computer or a target board

target board A Single Board Computer who can run Python or C++ code

Pattern As described in the book, this represents a graphical figure made by the leds in a defined matrix. Within a sequence it can be compared to a photo at a certain time.

Sequence A sequence is based on patterns, it's a list of patterns exactly. This is very handy when cascading patterns with different time speeds in between. Thus, you can define and send a pattern or different patterns one by one to make a complete sequence or animation.

Part Two



2. Presenting Information

2.1 Table

2.2 Citation

This statement requires citation [1]; this one is more specific [2, page 122]

Bibliography

Books

[Smi12] John Smith. *Book title*. 1st edition. Volume 3. 2. City: Publisher, Jan. 2012, pages 123–200 (cited on page 17).

Articles

[Smi13] James Smith. “Article title”. In: 14.6 (Mar. 2013), pages 1–8 (cited on page 17).

Index