



# PyRingo - Especificações de Requisitos REQ\_PR\_JogoRingo\_2021Fev19

Gabriel Carneiro  
Lorenzo Maturano  
Luiz Gustavo Saibro  
v1.4

9/5/2021

## Tabela de Versionamento

Versão	Autor(es)	Data	Ação
1.0	Gabriel Carneiro	19/02/2021	Construção do esqueleto da especificação de requisitos.
1.1	Luiz Gustavo Saibro	19/02/2021	Elaboração da especificação de requisitos
1.2	Lorenzo Maturano	25/02/2021	Alterações no modelo do documento e elaboração da especificação de requisitos.
1.3	Lorenzo Maturano	25/03/2021	Alterações de requisitos, conforme feedback do professor.
1.4	Gabriel Carneiro	09/05/2021	Correções nas definições do jogo e melhoria nas descrições dos movimentos.

# Sumário

<b>1</b>	<b>Introdução</b>	<b>4</b>
1.1	Objetivo . . . . .	4
1.2	Definições . . . . .	4
1.3	Referências . . . . .	4
<b>2</b>	<b>Regras do Jogo</b>	<b>5</b>
2.1	Estado Inicial do Tabuleiro . . . . .	5
2.2	Objetivo . . . . .	5
2.3	Movimentos . . . . .	5
<b>3</b>	<b>Visão Geral do Sistema</b>	<b>6</b>
3.1	Arquitetura do Software . . . . .	6
3.2	Premissas de Desenvolvimento . . . . .	6
<b>4</b>	<b>Requisitos da Aplicação</b>	<b>7</b>
4.1	Requisitos Funcionais . . . . .	7
4.2	Requisitos Não-Funcionais . . . . .	7
<b>5</b>	<b>Esboço da Interface Gráfica</b>	<b>8</b>

# 1 Introdução

## 1.1 Objetivo

Desenvolvimento de um software **stand-alone**, modelando o jogo de tabuleiro e estratégia **Ringo**, na linguagem de programação **Python**.

## 1.2 Definições

**Ringo** é um jogo de tabuleiro abstrato de estratégia e de dois jogadores.

- **Argola:** componente do jogo, manipulado pelo jogador;
- **Peça:** componente do jogo, manipulado pelo jogador;
- **Posição:** local do tabuleiro que pode estar vazio, ocupado por uma peça, por um anel ou por uma peça e um anel;
- **Posição Livre:** posição do tabuleiro não ocupada por argola ou peça;
- **Posição Ocupada:** posição do tabuleiro ocupada por argola ou peça;

O jogo é composto por **dois conjuntos de 10 peças e 4 argolas**, sendo cada conjunto com uma cor específica para identificar jogadores e um **tabuleiro** em formato de matriz, onde as peças do jogo são posicionadas.

## 1.3 Referências

- [https://www.youtube.com/watch?v=V10gy\\_BnaNk](https://www.youtube.com/watch?v=V10gy_BnaNk).
- <https://boardgamegeek.com/boardgame/261490/ringo/videos/all>.
- [https://www.youtube.com/watch?v=YHdRpfejvqQ&ab\\_channel=BoardGameGeek](https://www.youtube.com/watch?v=YHdRpfejvqQ&ab_channel=BoardGameGeek).

## 2 Regras do Jogo

### 2.1 Estado Inicial do Tabuleiro



Figura 1: Formação inicial do tabuleiro.

Um dos conjuntos de argolas ocupa as posições do canto, já o outro conjunto ocupa os espaços entre os cantos. Não é necessário manter o mesmo padrão de cores. Ou seja, em outra partida, os anéis brancos poderiam ocupar os cantos. As peças são inicialmente mantidos fora do tabuleiro.

### 2.2 Objetivo

Ganha o jogador que posicionar na vertical, na horizontal, ou na diagonal um conjunto de 4 peças ou 4 argolas consecutivos da sua cor.

### 2.3 Movimentos

Os movimentos podem ser divididos em dois momentos, quando o jogador ainda possui peças e quando todas suas peças já estão em jogo

- O jogador cujo o conjunto de argolas ocupam os cantos inicia o jogo.
- Se o jogador possui peças:
  - O jogador executa duas ações por rodada: **movimentar peça e movimentar argola**.

- O jogador inicia colocando uma peça **dentro de alguma argola livre**, não importando a cor da argola.
- Após colocar sua peça a uma argola livre, o jogador deve movimentar a **argola que foi ocupada**.
- A argola pode ser posicionado em uma **posição livre**, de maneira adjacente a outra argola ou peça, podendo ser tanto na vertical quanto na horizontal.
- Se todas peças estão em jogo:
  - Agora, antes de colocar uma peça no tabuleiro, **o jogador deve remover uma de suas peças que foi colocada no tabuleiro**.
  - Ao remover uma peça o jogador não pode segmentar o conjunto de peças e argolas em dois conjuntos separados.

## 3 Visão Geral do Sistema

### 3.1 Arquitetura do Software

Software **stand-alone**, modelando o jogo **Ringo**, utilizando a linguagem de programação **Python** e o paradigma de **Programação Orientado a Objetos**.

### 3.2 Premissas de Desenvolvimento

- Implementado em **Python**.
- Executável em qualquer ambiente com Python3.
- Script de instalação para criação de um ambiente virtual e instalação de dependências.
- Script de desinstalação.
- Interface gráfica.
- Modelagem UML utilizando o software **Visual Paradigm**.

## 4 Requisitos da Aplicação

### 4.1 Requisitos Funcionais

- **RF01 - Iniciar jogadores:** O programa deve possibilitar a criação de dois jogadores antes de iniciar um novo jogo, com seus respectivos nomes.
- **RF02 - Colocar peça na posição:** O programa deve possibilitar ao jogador inserir uma peça em seu turno em um local possível dentro das regras do jogo.
- **RF03 - Colocar argola na posição:** O programa deve possibilitar o jogador mudar a argola de posição em seu turno em um local possível, seguindo as regras do jogo.
- **RF04 - Remover peça de posição:** O programa deve possibilitar ao jogador remover uma peça de posição em seu turno em um local possível dentro das regras do jogo, quando não tiver mais peças disponíveis.
- **RF05 - Reiniciar partida:** O programa deve reiniciar a partida automaticamente após notificar o vencedor.

### 4.2 Requisitos Não-Funcionais

- **RNF01:** O jogo deve ser desenvolvido utilizando a linguagem de programação Python, versão 3.
- **RNF02:** O jogo deve utilizar a biblioteca PyGame para desenvolver a interface gráfica.
- **RNF03:** O jogo deve ter uma única interface para os dois jogadores.
- **RNF04:** Para controle de versionamento utilizar o github.

## 5 Esboço da Interface Gráfica

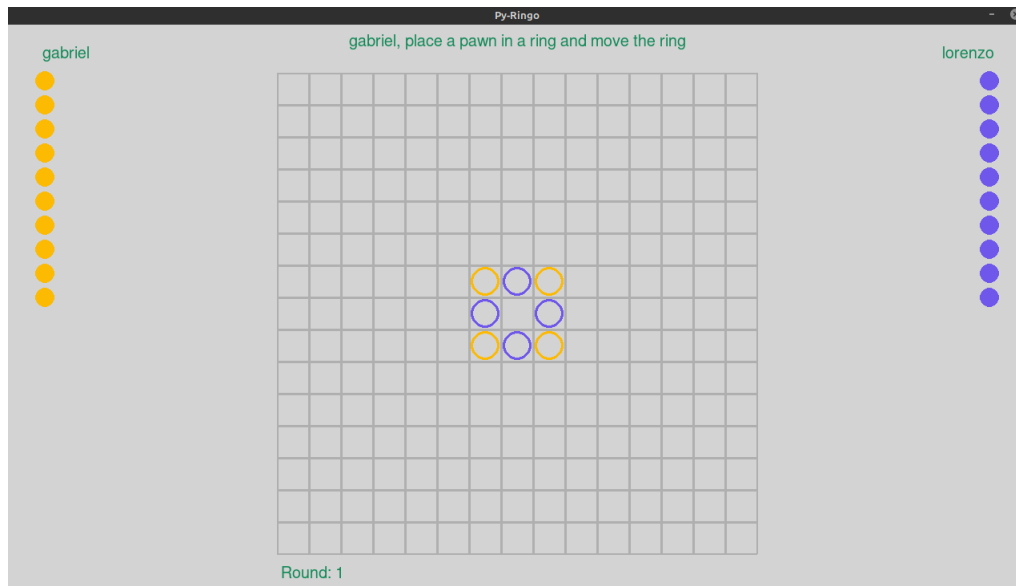


Figura 2: Esboço da Interface Gráfica