Falling in love with the DMG

Quentin "sunbro" Barbarat

GConfs

NDI 2020 - December 3rd 2020

What's a DMG?

What's a DMG?

"[...] You have to hold one in your hands. You study the finer details. The colors of the plastics, the textures, the sleek shine, it's firmness, it's weight, the craftsmanship that went into it's design. [...]

At this point, if you're not already in love, you may need to get a pulse check, because the DMG is the alpha, the omega, the be all, end all, the big bang, the mother womb, the warm caressing light of eternity.

DMG is nirvana, bliss and eutopia, all rolled into a portable, plastic miniature monolith that stands the test of time."

— u/the_8bit_kingdom

What's a DMG?

What's a DMG?

Figure: Perfection

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F
  2. 0x1: #306230

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F
  2. 0x1: #306230
  3. 0x2: #8BAC0F

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F
  2. 0x1: #306230
  3. 0x2: #8BAC0F
  4. 0x3: #9BBC0F

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F
  2. 0x1: #306230
  3. 0x2: #8BAC0F
  4. 0x3: #9BBC0F
- Custom 8-bit Sharp LR35902

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F
  2. 0x1: #306230
  3. 0x2: #8BAC0F
  4. 0x3: #9BBC0F
- Custom 8-bit Sharp LR35902
- 4-way D-pad

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F
  2. 0x1: #306230
  3. 0x2: #8BAC0F
  4. 0x3: #9BBC0F
- Custom 8-bit Sharp LR35902
- 4-way D-pad
- 4 buttons: A, B, select, start

What's a DMG?

Figure: Perfection

- 160 x 144 LCD screen
- 1 x 4 color palette
  1. 0x0: #0F380F
  2. 0x1: #306230
  3. 0x2: #8BAC0F
  4. 0x3: #9BBC0F
- Custom 8-bit Sharp LR35902
- 4-way D-pad
- 4 buttons: A, B, select, start
- 8 KB of RAM & VRAM

Haven't you done this before?

Haven't you done this before?

- Yes I have

Haven't you done this before?

- Yes I have
- I wanted to learn more

Haven't you done this before?

- Yes I have
- I wanted to learn more
- I'm still no expert

Why are you doing this to yourself?

It's all because of Hacktoberfest ...

Why make a gameboy game in 2020

Why make a gameboy game in 2020

- Creative problem solving due to all the constraints

Why make a gameboy game in 2020

- Creative problem solving due to all the constraints
- An active community, on modern social networks

Why make a gameboy game in 2020

- Creative problem solving due to all the constraints
- An active community, on modern social networks
- Lots of community-driven documentation

Why make a gameboy game in 2020

- Creative problem solving due to all the constraints
- An active community, on modern social networks
- Lots of community-driven documentation
- Easy-to-grasp graphics pipeline

Why make a gameboy game in 2020

- Creative problem solving due to all the constraints
- An active community, on modern social networks
- Lots of community-driven documentation
- Easy-to-grasp graphics pipeline
- Minimal tooling needed

What you need to get started

Sidenote: what about the OS?

Sidenote: what about the OS?

- Most tools work on typical GNU+Linux distributions

Sidenote: what about the OS?

- Most tools work on typical GNU+Linux distributions
- One tool needs WINE, but works flawlessly

Sidenote: what about the OS?

- Most tools work on typical GNU+Linux distributions
- One tool needs WINE, but works flawlessly
- Windows 10 + Ubuntu WSL 2 was my daily driver

What you need to get started

What you need to get started

- A development toolkit
- A code editor
- A sprite editor
- A tile data generator
- An emulator
- Documentation

Tools - Development toolkit: RGBDS

Tools - Development toolkit: RGBDS

- Assembler, linker, fixer, and image converter

Tools - Development toolkit: RGBDS

- Assembler, linker, fixer, and image converter
- Uses GBZ80 assembly

Tools - Development toolkit: RGBDS

- Assembler, linker, fixer, and image converter
- Uses GBZ80 assembly
- Provides useful debugging options

Tools - Development toolkit: RGBDS

- Assembler, linker, fixer, and image converter
- Uses GBZ80 assembly
- Provides useful debugging options
- Actively maintained

Tools - Development toolkit: RGBDS

- Assembler, linker, fixer, and image converter
- Uses GBZ80 assembly
- Provides useful debugging options
- Actively maintained
- Very well documented

Tools - Code editor: VS Code

Tools - Code editor: VS Code

- “Lightweight” & modern

Tools - Code editor: VS Code

- “Lightweight” & modern
- Integrated terminal

Tools - Code editor: VS Code

- “Lightweight” & modern
- Integrated terminal
- Syntax highlighting plugin

Tools - Code editor: VS Code

- “Lightweight” & modern
- Integrated terminal
- Syntax highlighting plugin

Your favorite code editor probably has a syntax highlighting plugin too - you can check the list here

Tools - Sprite editor: Aseprite

Tools - Sprite editor: Aseprite

- Open source, but available for $19.99

Tools - Sprite editor: Aseprite

- Open source, but available for $19.99
- Can be used with custom palettes

Tools - Sprite editor: Aseprite

- Open source, but available for $19.99
- Can be used with custom palettes
- Has an indexed mode

Tools - Sprite editor: Aseprite

- Open source, but available for $19.99
- Can be used with custom palettes
- Has an indexed mode
- Very user friendly

Tools - Tile data generator: GBTDG

Tools - Tile data generator: GBTDG

- Open source, but not maintained

Tools - Tile data generator: GBTDG

- Open source, but not maintained
- Written in HTML5 + JS

Tools - Tile data generator: GBTDG

- Open source, but not maintained
- Written in HTML5 + JS
- No external dependencies

Tools - Tile data generator: GBTDG

- Open source, but not maintained
- Written in HTML5 + JS
- No external dependencies
- Easy to use, with useful options

Tools - Emulator: BGB

Tools - Emulator: BGB

- Made for Windows™, but works flawlessly with WINE

Tools - Emulator: BGB

- Made for Windows™, but works flawlessly with WINE
- La crème de la crème

Tools - Emulator: BGB

- Made for Windows™, but works flawlessly with WINE
- La crème de la crème
- Hardware visualization

Tools - Emulator: BGB

- Made for Windows™, but works flawlessly with WINE
- La crème de la crème
- Hardware visualization
- Integrated assembly debugger

Tools - Documentation: gbdev.io

Tools - Documentation: gbdev.io

- Curated

Tools - Documentation: gbdev.io

- Curated
- Non-exhaustive

Tools - Documentation: gbdev.io

- Curated
- Non-exhaustive
- Probably has what you're looking for

Sidenote: about gingerbread

Sidenote: about gingerbread

- Has a 70-page companion book

Sidenote: about gingerbread

- Has a 70-page companion book
- Not "optimized"

Sidenote: about gingerbread

- Has a 70-page companion book
- Not "optimized"
- Does not work with the latest version of RGBDS

Sidenote: about gingerbread

- Has a 70-page companion book
- Not "optimized"
- Does not work with the latest version of RGBDS

You should still read *Game Boy Assembly Programming for the Modern Game Developer* if you don't plan on using Gingerbread.

Gameboy specific constraints

Gameboy specific constraints

- 3 layers

Gameboy specific constraints

- 3 layers
  - Background

Gameboy specific constraints

- 3 layers
  - Background
  - Window

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles
- 256 items in tilemap

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles
- 256 items in tilemap
- 32 KB per rom

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles
- 256 items in tilemap
- 32 KB per rom
- 10 sprites per line

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles
- 256 items in tilemap
- 32 KB per rom
- 10 sprites per line

Figure: 10 sprites on a line

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles
- 256 items in tilemap
- 32 KB per rom
- 10 sprites per line

Figure: More than 10 sprites on a line

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles
- 256 items in tilemap
- 32 KB per rom
- 10 sprites per line
- Essentially no 3D

Gameboy specific constraints

- 3 layers
  - Background
  - Window
  - Sprite
- 20 x 18 8x8 tiles
- 256 items in tilemap
- 32 KB per rom
- 10 sprites per line
- Essentially no 3D

Figure: Toy Story Racer - 2001

Show us the (op)code(s)!

Show us the (op)code(s)!

- Just enough to get to know them

Show us the (op)code(s)!

- Just enough to get to know them
- Some “usual” operations are missing

Show us the (op)code(s)!

- Just enough to get to know them
- Some “usual” operations are missing
- Macros & functions

Show us the (op)code(s)!

- Just enough to get to know them
- Some “usual” operations are missing
- Macros & functions
- Global variables

Show us the (op)code(s)!

- Just enough to get to know them
- Some “usual” operations are missing
- Macros & functions
- Global variables

A *hardware.inc* or equivalent is highly recommended

Show us the workflow!

Show us the workflow!

Let's add an NPC interaction!

Show us the workflow!

Let's add an NPC interaction!

1. Choose sprite

Show us the workflow!

Let's add an NPC interaction!

1. Choose sprite
2. Add to background

Show us the workflow!

Let's add an NPC interaction!

1. Choose sprite
2. Add to background
3. Generate tile data & update

Show us the workflow!

Let's add an NPC interaction!

1. Choose sprite
2. Add to background
3. Generate tile data & update
4. Add dialogue string

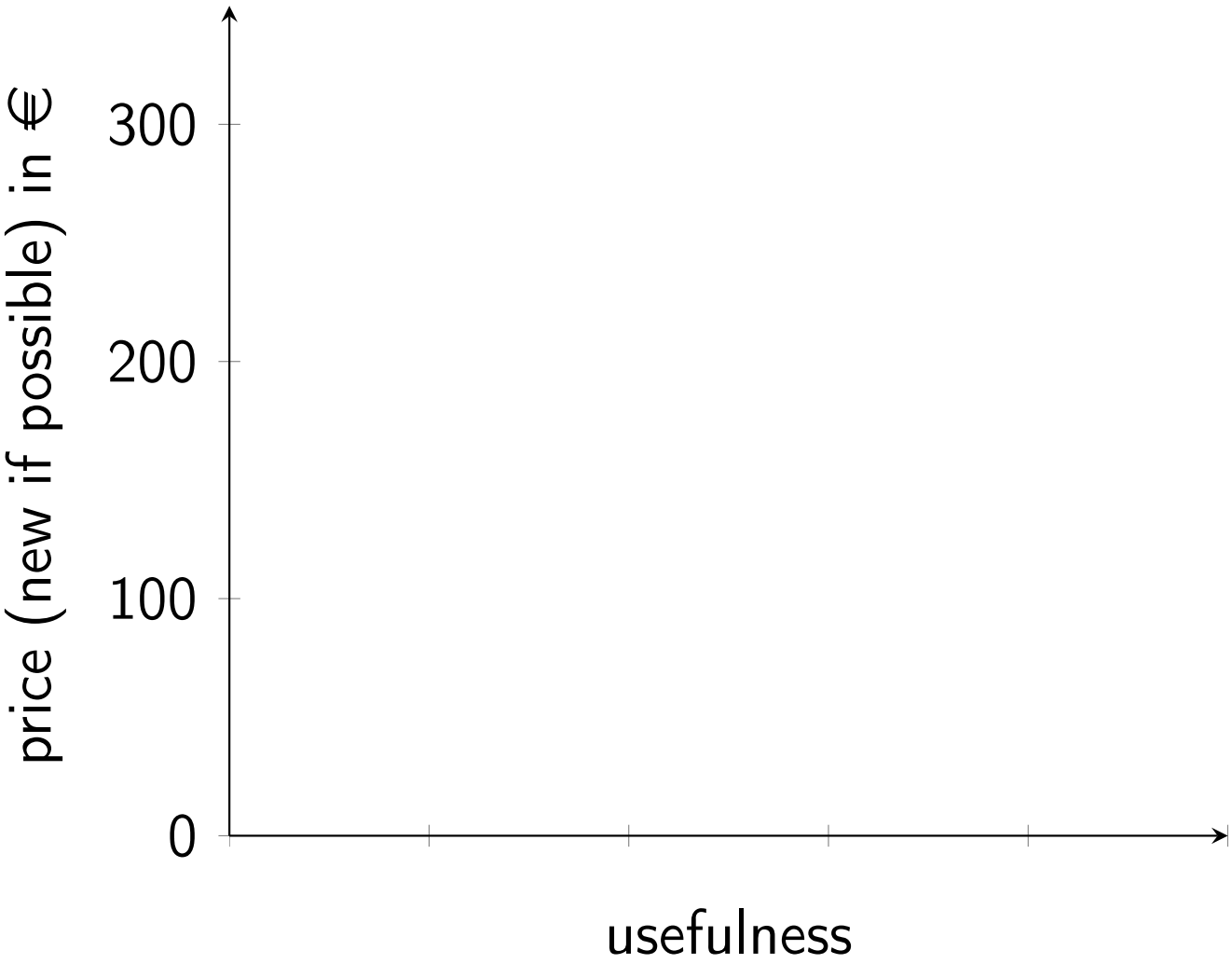
Show us the workflow!

Let's add an NPC interaction!

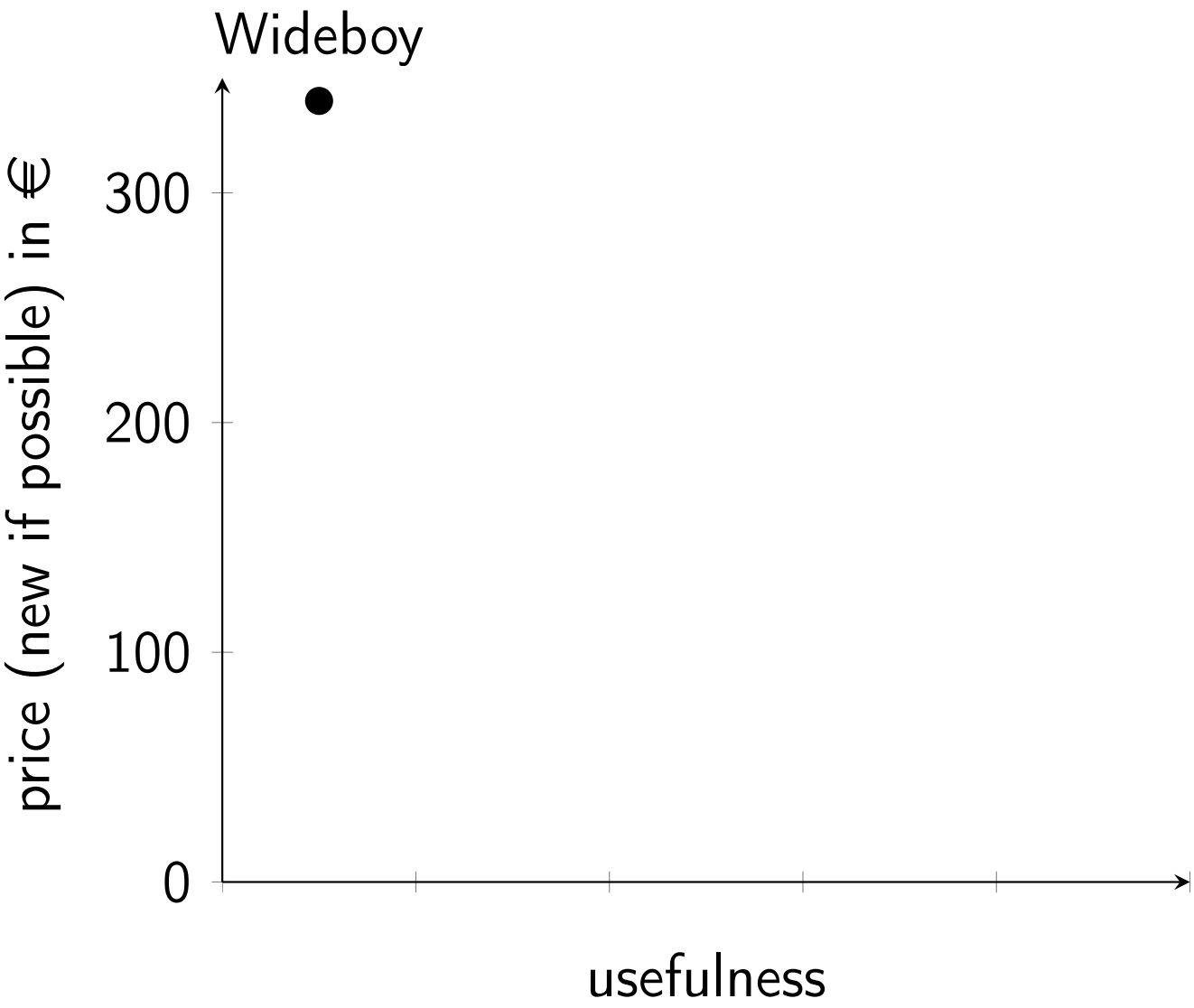
1. Choose sprite
2. Add to background
3. Generate tile data & update
4. Add dialogue string
5. Add interaction to list

Playing your game

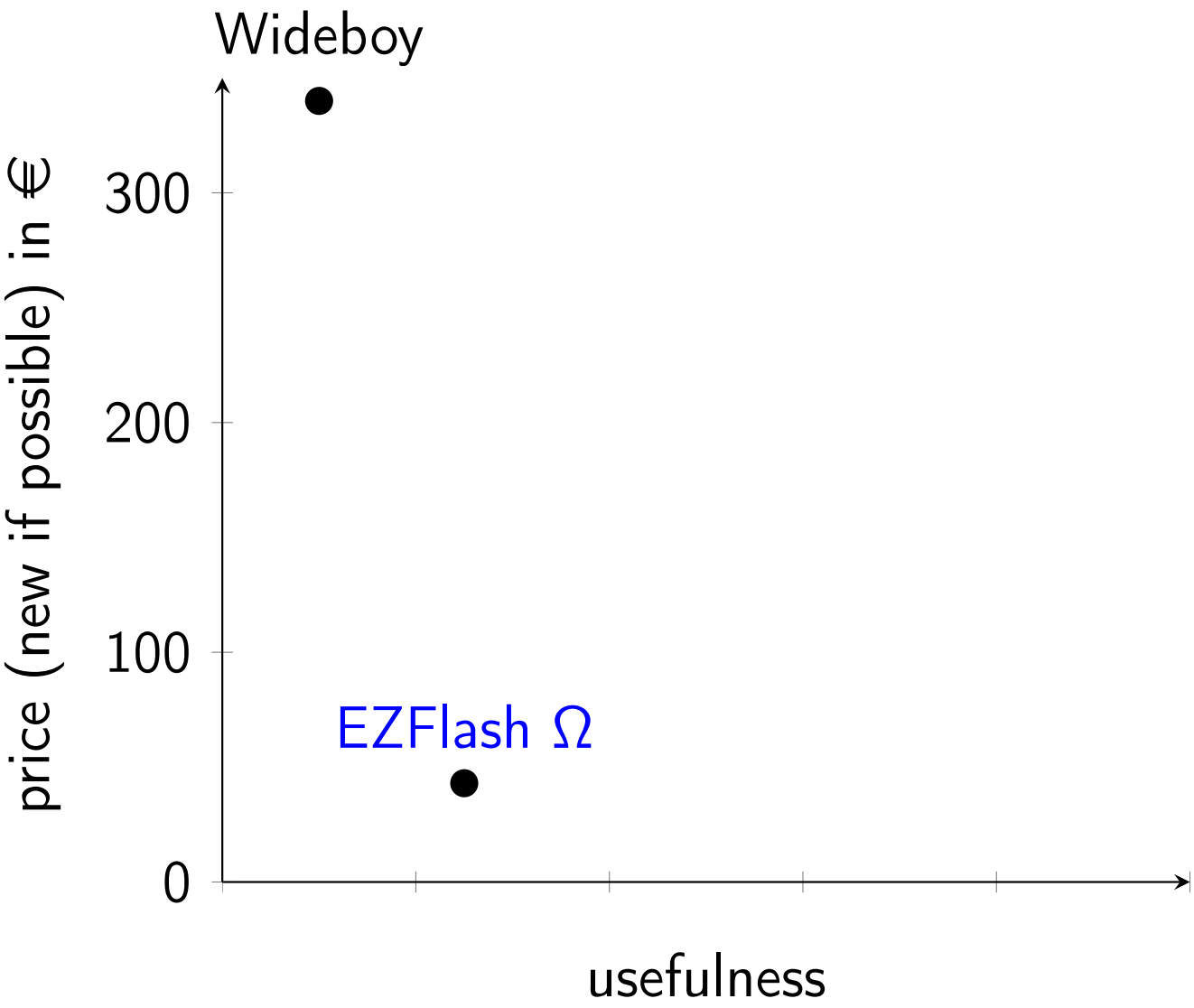
Playing your game



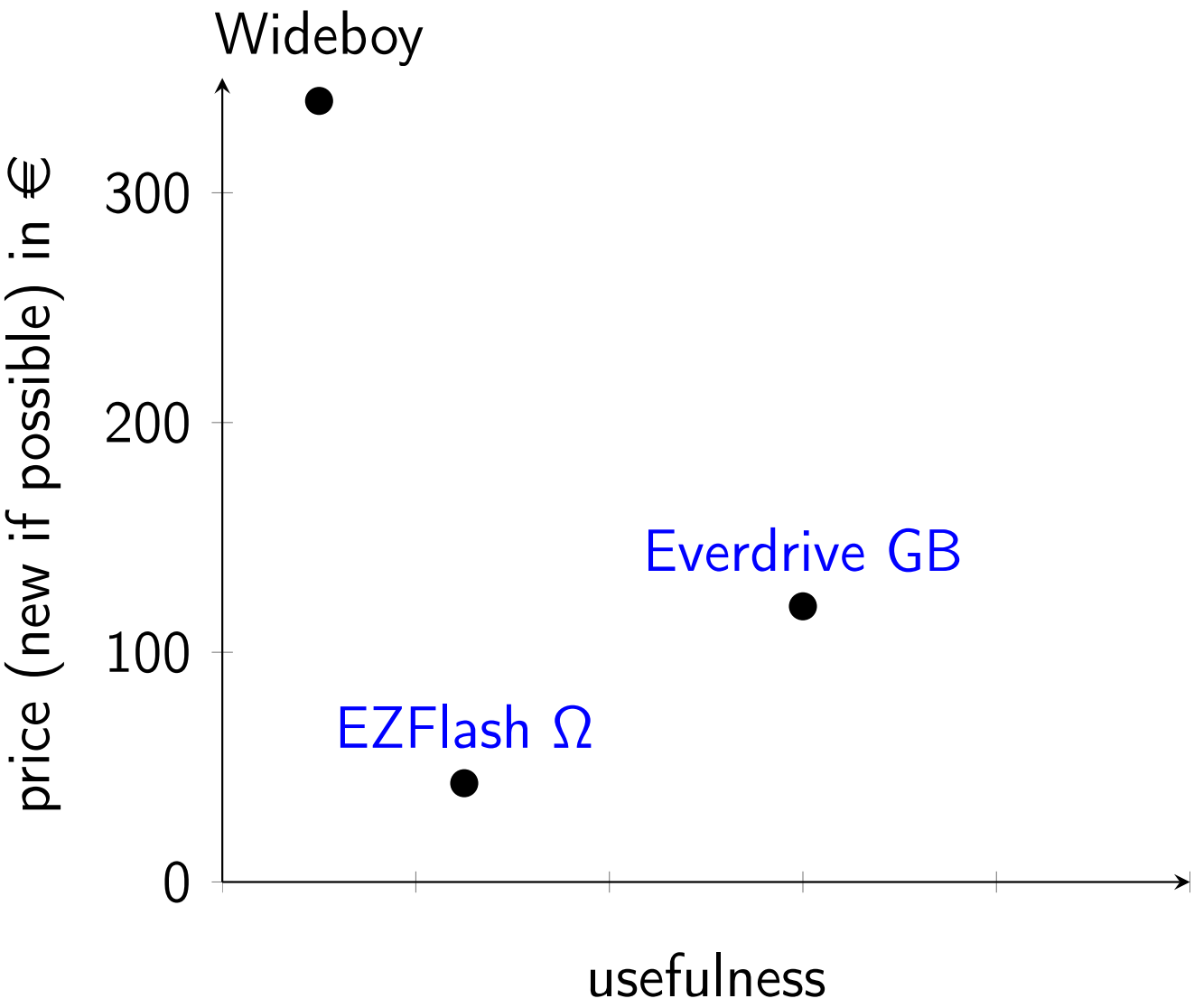
Playing your game



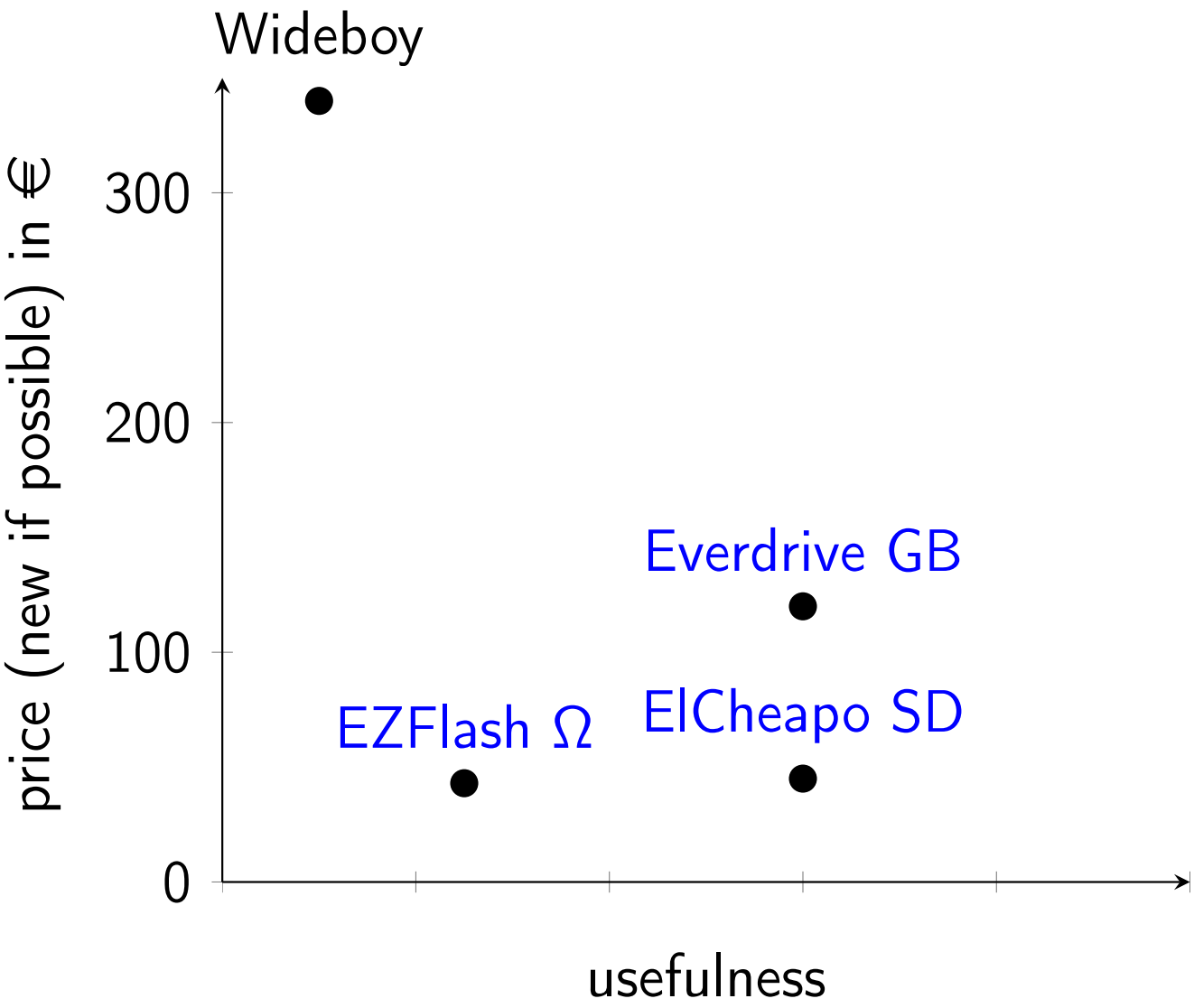
Playing your game



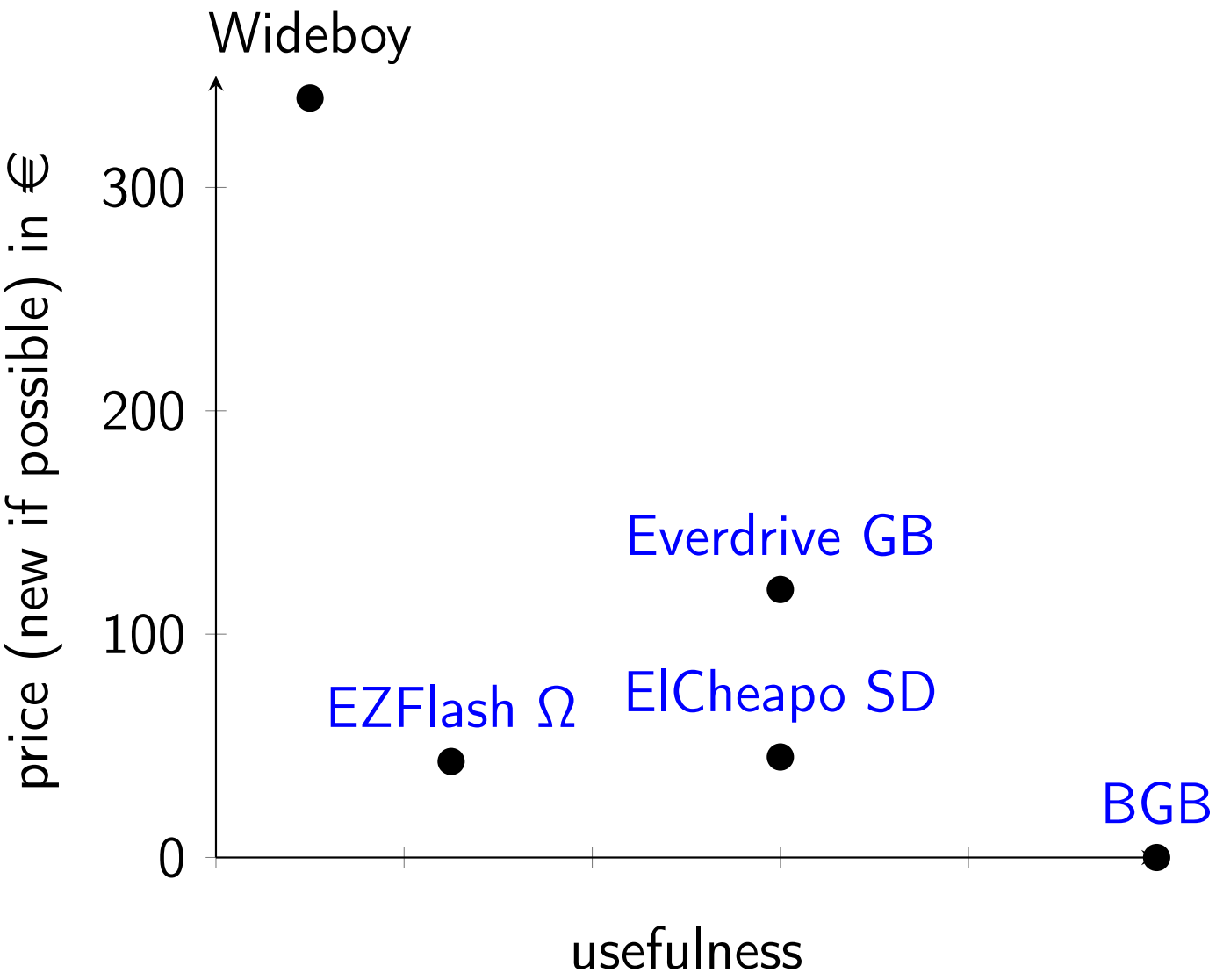
Playing your game



Playing your game



Playing your game



Quick tips

Quick tips

- Don't forget to *ret* !

Quick tips

- Don't forget to *ret* !
- Try to understand other peoples' code

Quick tips

- Don't forget to *ret* !
- Try to understand other peoples' code
- Don't hesitate to take shortcuts at first

Why I love the DMG

Why I love the DMG

- “Simple” graphics code compared to OpenGL / SDL

Why I love the DMG

- “Simple” graphics code compared to OpenGL / SDL
- Minimal toolkit and footprint compared to famous engines

Why I love the DMG

- “Simple” graphics code compared to OpenGL / SDL
- Minimal toolkit and footprint compared to famous engines
- Nostalgia and ever growing respect for the OG

What the future holds

What the future holds

- Add objectives

What the future holds

- Add objectives
- Add (real) menus

What the future holds

- Add objectives
- Add (real) menus
- Add sound

What the future holds

- Add objectives
- Add (real) menus
- Add sound
- Move on to GBA (thanks to butano)

Questions, comments, something unclear ?

Thank you for listening

Useful links - Tools

1. Development toolkit - RGBDS
2. Code editor - VS Code
3. Sprite editor - Aseprite
4. Tile data generator - GBTDG
5. Emulator - BGB
6. Constant definitions - hardware.inc

Useful links - Documentation

1. Gingerbread companion book - Game Boy Assembly Programming for the Modern Game Developer
2. Useful links list - Awesome Game Boy Development
3. Game Boy technical reference - Pan Docs
4. GBZ80 Opcode reference - RGBDS GBZ80 opcode reference
5. GBZ80 Instruction set - Optables
6. Source code - GitHub
7. Slides - sunbro.dev/talks