GNOME Crosswords

Jonathan Blandford <jrb@gnome.org>

https://gitlab.gnome.org/jrb/crosswords
http://libipuz.org/

Agenda

- **GNOME Crosswords**
- **Writing** Crosswords
- Writing Crossword **Writers**

Act 1

GNOME Crosswords

What is GNOME Crosswords?

GNOME Crosswords is a **project** for the **GNOME/Linux** ecosystem

Started this four years ago:

- I needed a new hobby
- I'm a passionate crossword enthusiast:
  - Started doing **Times of London cryptic crosswords** as a kid with my parents and grandparents
  - Now I do them with my family!
- I had a vision in my head

What is GNOME Crosswords? — *By the numbers*

- Code at gitlab.gnome.org/jrb/
  - **85KLOC total.** 60 KLOC in the app and 25 KLOC in the library
  - **27K words** of design docs (development guide)
  - **6 translations** (Dutch, English, German, Italian, Portuguese, and Spanish)
  - **126 test sections**
  - **33 different contributors**
    - (I'm **~82%** of the total **commits** — a relatively healthy FOSS project!)
    - **6** different Google Summer of Code / Outreachy **interns**

Special thanks to **Federico**, **Tanmay**, **Davide**, **Philip**, and **Vinson** who help me maintain this!

What is GNOME Crosswords?

- `libipuz` — a **shared library** for loading, mutating, and saving puzzles
- `Crosswords` — a standalone ***game*** for crossword-style puzzles
- `Crossword Editor` — an ***authoring tool*** for writing puzzles

What is GNOME Crosswords?

- **`libipuz`** is the underlying library supporting GNOME Crosswords
  - `libipuz` loads, saves, and manipulates `.ipuz` puzzle files.
    - **Manipulation** is the key differentiator of this library
  - Supports eight different puzzle kinds
  - Extensively documented at https://libipuz.org/
  - Written in C, though we are migrating parts to Rust

Thanks Pranjal and GSoC!

  - Uses **glib**, **GObject**, and ***gobject-introspection***:
    - Provides a modern C experience
    - gobject-introspection can export a library to over 25 different language bindings

What is GNOME Crosswords?

- Written using GTK4 / libadwaita and distributed on Linux through Flathub
  - *Snap / RPM / Arch as well!*
- UI code shared between the game and editor
  - Over 100 custom widgets across both apps
  - The editor is a **lot** more complex than the game
- Community has built it on Mac and Windows

There's a community provided .apk as well!

Demo

The Game

Act 2

Writing Crosswords

How do you write a crossword?

1. Pick a theme (optional)
2. Create a grid
3. Write the clues

With standard (American) crosswords, creating a grid is really hard

On the other hand, writing clues is tricky for cryptic crosswords

A fundamental rule of crosswords:

Every cell *needs at least* ***two paths*** *to its answer*

*These two paths means it is "**checked**"*

Standard and Cryptic crosswords satisfy this rule in different ways

Crossword setters use the term ***unch*** — short for *"unchecked"* — to describe when there is only one path to answer a cell.

Unches are to be avoided.

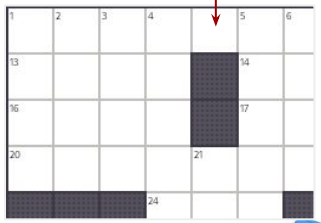
Standard Crosswords

1 U	2 N	3 D	4 E	5 R	■	■	6 M	7 I	8 L	9 D	■	10 B	11 C	12 C
13 P	E	E	L	E	■	14 A	E	R	I	E	■	15 A	H	H
16 T	R	A	F	F	17 I	C	L	A	N	E	■	18 L	O	U
19 O	D	D	■	■	20 L	A	O	■	■	21 M	22 U	S	I	C
■	■	23 W	24 A	25 L	K	I	N	26 T	27 H	E	P	A	R	K
■	28 T	R	U	E	■	■	■	29 R	E	D	S	■	■	■
30 M	O	O	L	A	■	31 A	32 J	A	R	■	■	33 P	34 T	35 A
36 H	U	N	D	R	37 E	D	A	C	R	38 E	39 W	O	O	D
40 O	R	G	■	■	41 S	O	N	Y	■	42 M	I	L	E	S
■	■	■	43 A	44 S	A	P	■	■	■	45 I	P	A	D	■
46 K	47 A	48 B	U	K	I	T	49 H	50 E	51 A	T	E	R	■	■
52 A	L	E	X	I	■	■	53 A	N	T	■	■	54 O	55 A	56 T
57 R	I	A	■	58 F	59 O	60 U	N	D	F	61 A	62 M	I	L	Y
63 M	A	C	■	64 F	U	M	E	S	■	65 N	A	D	I	R
66 A	S	H	■	67 S	I	P	S	■	■	68 I	N	S	T	A

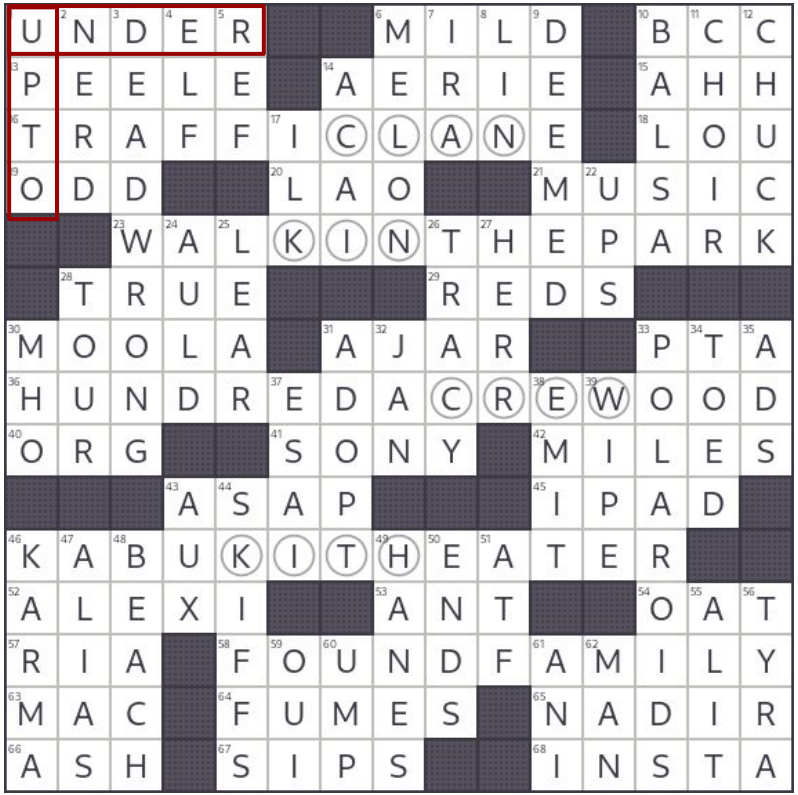
L. A. Times – Jul 3, 2024

Every cell has an across and down clue to let you know you have the right answer.

UNCH: Avoid this pattern



Standard Crosswords



L. A. Times – Jul 3, 2024

Assorted clues:

1ac. Beneath
1dn. As many as

The answer to 1ac could be "UNDER" or "BELOW"
We need to try to solve 1dn to be sure

Standard Crosswords — Puzzle theme

L. A. Times – Jul 3, 2024

Assorted clues:

58ac. Closely knit community that provides social support, or what 16, 23, 36, and 46 across contain?

FOUNDFAMILY describes the words in circles

Standard Crosswords — Tools

L. A. Times – Jul 3, 2024

Computer aided grid creation:

- **Automatic grid fill**
- **Word suggestion**
- **Word definitions**
- **Frequency lists**
- Well-curated **word list**
- Grid scores
- Hot spot detection
- …

Items in bold implemented

Cryptic Crosswords — Clues

1 T	R	2 U	S	3 T	Y	■	4 U	5 P	I	6 N	A	7 R	M	8 S
A	■	N	■	H	■	9 P	■	A	■	E	■	E	■	I
10 N	E	F	A	R	I	O	U	S	■	11 P	U	P	I	L
G	■	I	■	O	■	W	■	S	■	H	■	R	■	E
12 L	E	T	S	T	H	E	S	I	D	E	D	O	W	N
E	■	■	■	T	■	R	■	N	■	W	■	B	■	C
13 S	A	14 D	D	L	E	B	A	G	S	■	15 W	A	G	E
■	■	I	■	E	■	R	■	T	■	16 E	■	T	■	■
17 M	O	V	E	■	18 L	E	C	H	W	A	L	E	S	19 A
A	■	E	■	20 M	■	A	■	R	■	R	■	■	■	G
21 C	A	R	N	A	L	K	N	O	W	L	E	22 D	G	E
B	■	S	■	N	■	F	■	U	■	I	■	R	■	L
23 E	X	I	S	T	■	24 A	G	G	R	E	G	A	T	E
T	■	O	■	R	■	S	■	H	■	S	■	W	■	S
25 H	E	N	P	A	R	T	Y	■	26 S	T	I	L	T	S

Times of London – Jun 30, 2024

With cryptics, **clues** are used to provide redundancy. Every clue has two parts: a **straight definition** and some **wordplay**.

Cryptic Crosswords — Basic Clues

Bird | seen in the museum (3) | E | M | U | Hidden Word Clue

DEFINITION WORDPLAY

Aunt butchered | fish (4) | T | U | N | A | Anagram Clue

WORDPLAY DEFINITION

Cryptic Crosswords — Harder Clues

Article | that halved | troubles | in the colonies? (3-5) | A N T-H I L L S | Charade Clue

WORDPLAY | DEFINITION

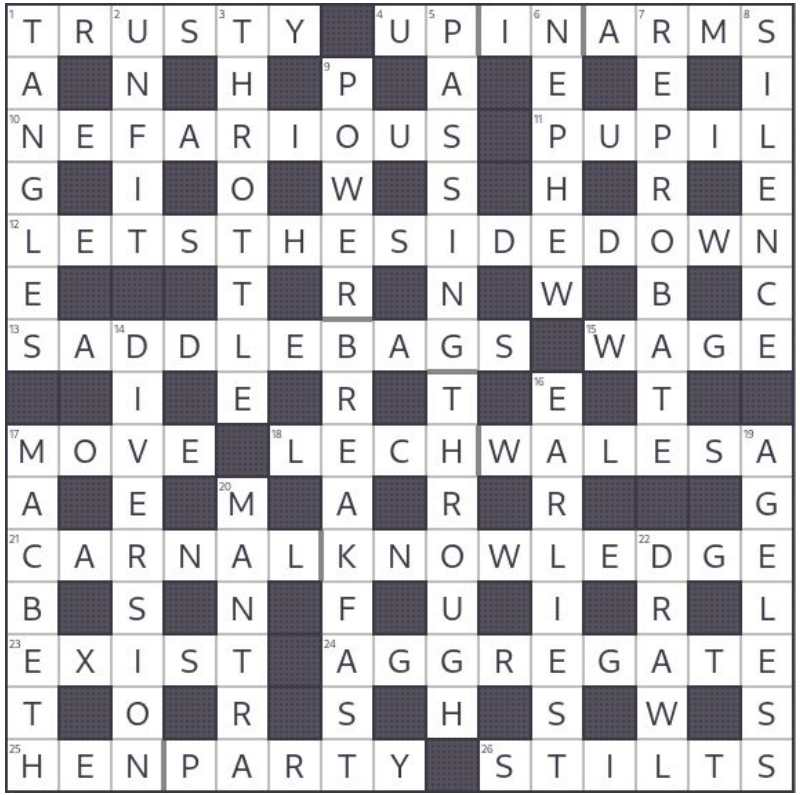
Last part of Aida | polished off, | get going (8) | A C T I V A T E | Misdirection!

WORDPLAY | DEFINITION

That's it for cryptic clues

Don't worry if this was confusing. Gemini is terrible at these, too!

Cryptic Crosswords — tools



Times of London – Jun 30, 2024

Computer aided clue authoring:

- **Anagrams / Cycles**
- **Alternations (odd/even)**
- **Deletions**
- **Word definitions**
- Hidden words
- Frequency lists
- Common sounds (homophones and spoonerisms)
- etc.

Items in bold implemented

Challenges writing crossword software

- Crosswords render like fonts
  - They don't scale linearly; grids need pixel-alignment
  - They fight with most toolkit's sizing assumptions
- Good puzzles follow conventions; great puzzles break them
  - Rebus clues, hidden answers/circles, non-rectangular clues, etc
  - We don't want to hard-code common conventions
- We need specialized tools to avoid those *unches*
  - Those tools need to be ***fast…***
  - Rapid experimentation is key to producing a great crossword
- Small edits can lead to big changes
  - Undo/Redo has to be central to the editor

Example: Small edits can lead to big changes in the puzzle

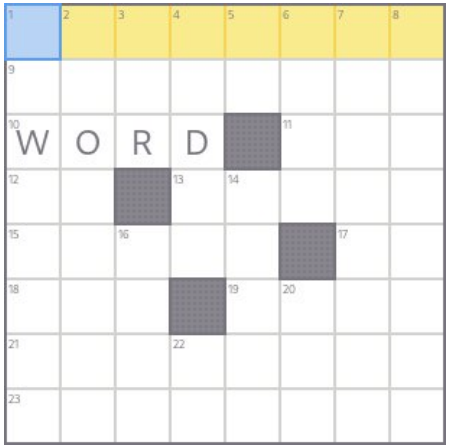
The philosophy of libipuz is to let the developer make any changes they want to a puzzle, even if it results in a non-standard puzzle.

Developers call the `_fix()` category of functions in order to enforce well-known heuristics.

Example code: Ipuz – 1.0: Intro to libipuz

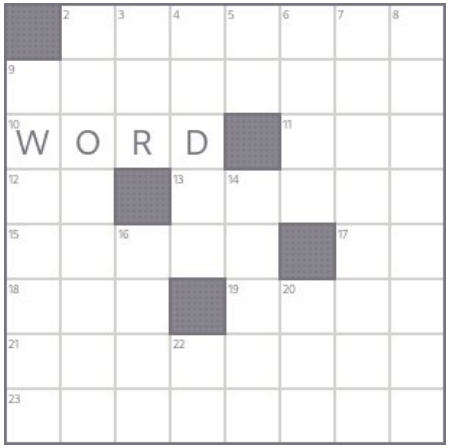
Example: Small edits can lead to big changes in the puzzle

Change (0,0) to a BLOCK



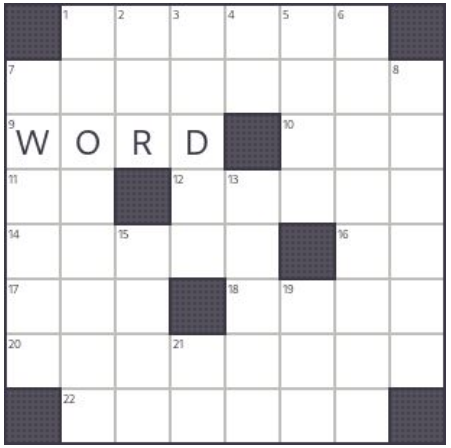
User presses the '#' key

Initial result



Numbering is wrong
Symmetry isn't maintained
Enumerations are broken

What we want



Clues are different
Styles could be different

Demo

The Editor

Act 3

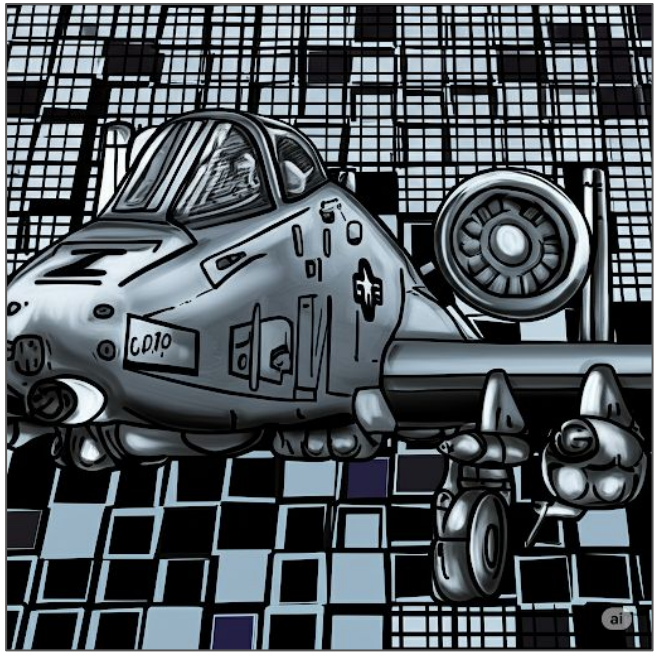
Writing Crossword Writers

Writing the Editor:
Requirements

- Must be ***useful*** to our users
- Simple to ***reason*** about the codebase
- Core operations must be ***testable***
- Assistants should be ***fast*** and easy to use
- ***Undo*** and ***redo*** are key!

Undo and redo are key

- Every user operation should be safe
- Structure the editor around the undo/redo code rather than treat it as an add-on



A Quick Tangent

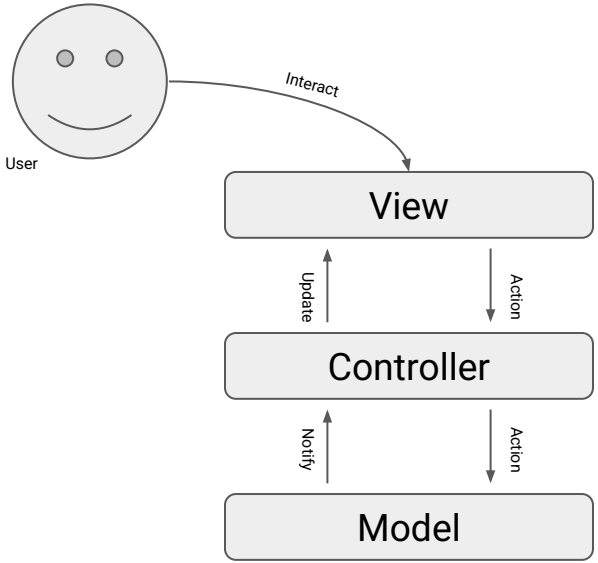
Desktop application development

Writing a Desktop application is a little different

- Largely in-process
  - Threads for long-running operations
- Main loop with events, signals, and callbacks
- Supports hugely variable monitor sizes
- Multiple inputs: touch, keyboard, and mouse are all equally valid
- 17msec target per frame (@60 fpsec)

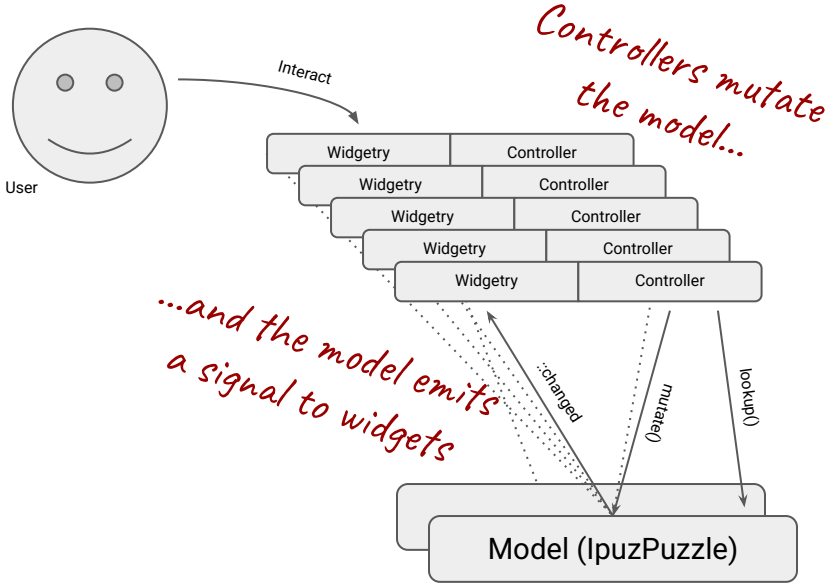
- **Users!**

MVC: Traditional Desktop Application Architecture



Source: Hundreds of random computer opinion websites

MVC: Traditional Desktop Application Architecture — *reality*



Advantages:

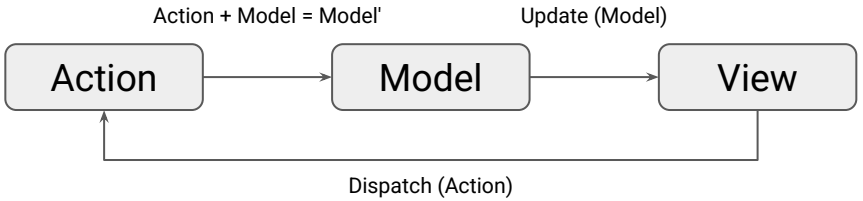
- Well understood; traditional
- Centralization of data
- Isolation of components
- ...

Disadvantages:

- State is spread across a graph of objects
- Undo/Redo is tricky
- Complicated ::changed signals.
- Widgets get out of sync, x-query each other
- Testing is hard; UI functions as a single unit

Action-Model-View: Unidirectional Data Flow Architecture

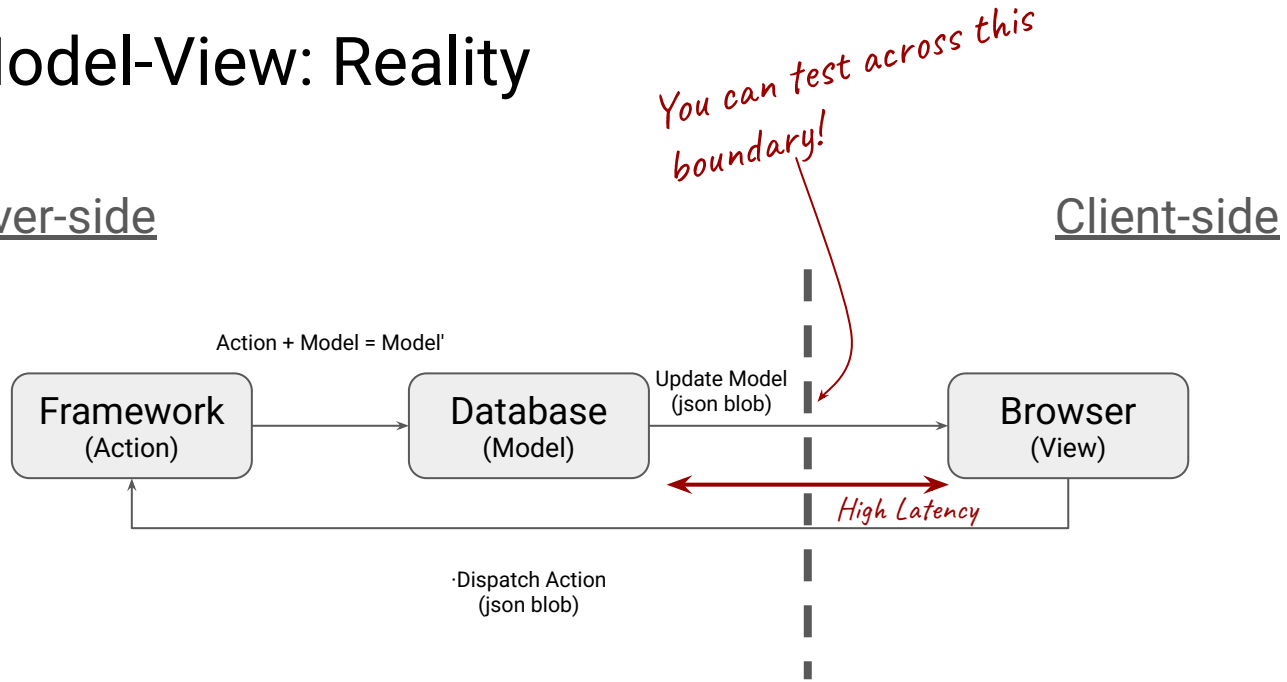
- UI callbacks don’t modify the model directly; they generate actions which are value-typed commands



- View is reconstructed from a value from the model

- Used to minimize round-trips to and from the view!

Action-Model-View: Reality



*This approach is highly **testable** and **predictable**. Actions and results can be replayed*

Writing the Editor:

Implementation rules

Converting Action-Model-View to a GTK Desktop App

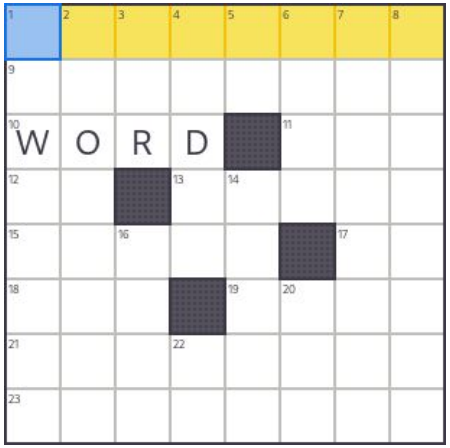
1. All state is kept in a single ***value***. Updates are **discrete** and **atomic**.

2. Widgets are **stateless** and **commutative**

3. All control flow is **centralized** in a **top-down dispatch** fashion. Widgets don't query or mutate the model

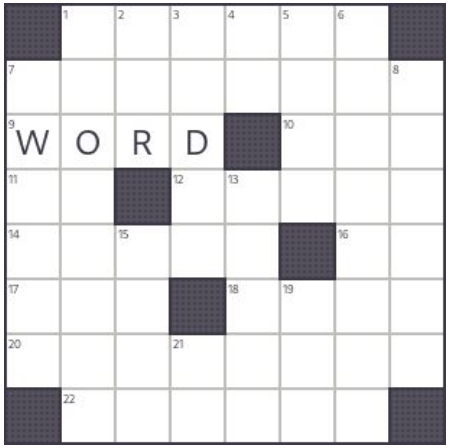
4. **Complexity** is **concentrated**

Action-Model-View: An example



State 1

User presses the '#' key



State 2

Point 1: *All state is kept in a single struct. Updates are discrete*

- **For the game**, all state is kept in a simple `GridState` struct
  - It includes mixture of semantic and display data
- **`For the editor,`** `EditState` includes over twenty five members including four distinct `GridStates`

States need to be representable as a ***value***

Every state has its own copy of the puzzle object

```
typedef struct
{
  IpuzCrossword *xword;
  GridStateBehavior behavior;

  /* Current keyboard state */
  IpuzCellCoord cursor;
  IpuzClueId clue;

  /* Modifiers */
  GridRevealMode reveal_mode;
  CrosswordsQuirks *quirks;
} GridState;
```


Point 1: *All state is kept in a single struct. Updates are discrete*

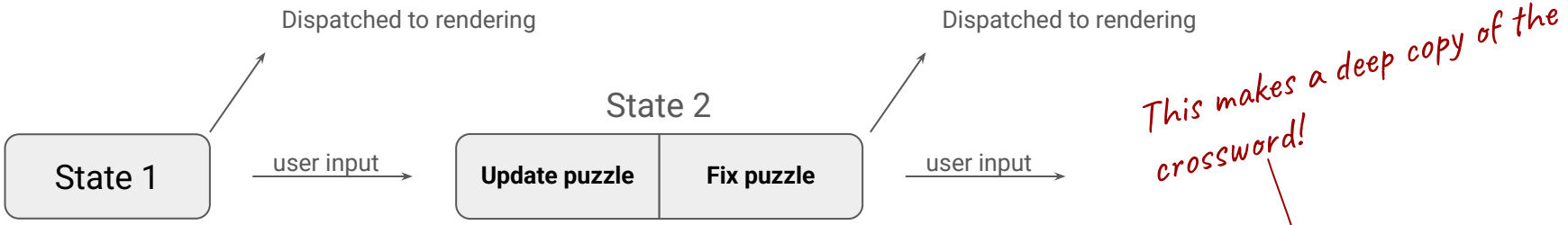
This makes a deep copy of the crossword!

User presses '#' key

state changes from State1 to State2

```
state_2 = grid_state_guess (self->state,
                            self->state->cursor,
                            "#"));
grid_state_unref (self->state);
self->state = state_2;
update_all (self);
```


Point 1: *All state is kept in a single struct. Updates are discrete*

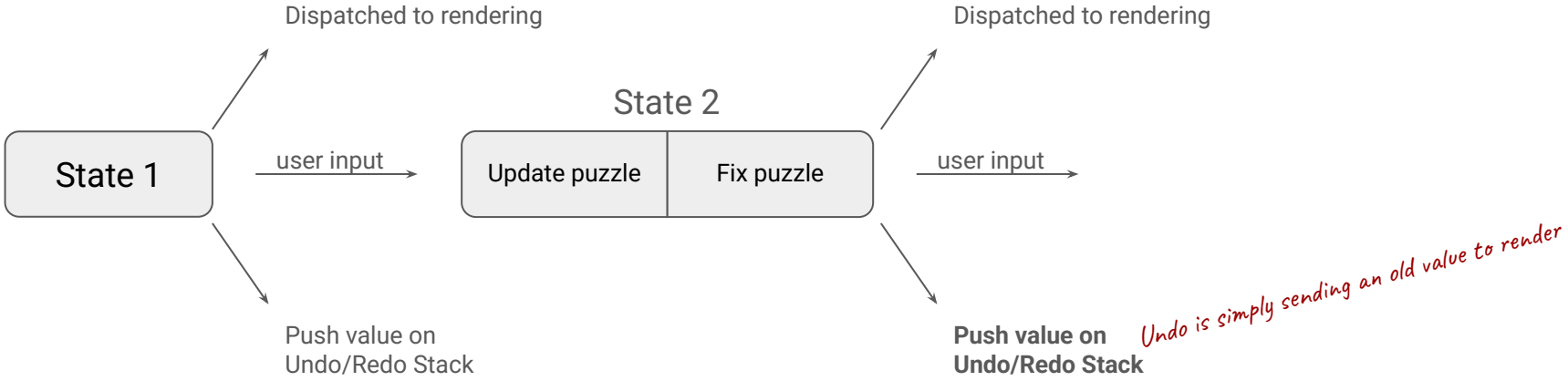


This makes a deep copy of the crossword!

When changing the puzzle:
1. First make all changes to it
2. Then fix the puzzle to follow convention

```
GridState *
grid_state_guess (GridState     *state,
                  IpuzCellCoord *coord,
                  const gchar   *guess)
{
  GridState *state = grid_state_clone (state);
  IpuzCell *cell =
    ipuz_crossword_get_cell (state->xword, coord); // coord is (0, 0)
  ipuz_cell_set_cell_type (cell, IPUZ_CELL_BLOCK);
  ipuz_crossword_fix_symmetry (state->xword,
                               IPUZ_SYMMETRY_ROTATIONAL_QUARTER,
                               coord);
  ipuz_crossword_fix_numbering (state->xword);
  ipuz_crossword_fix_clues (state->xword);
  ipuz_crossword_fix_enumerations (state->xword);
  …
```

Point 1: *All state is kept in a single struct. Updates are discrete*



Every undoable state is pushed to an Undo/Redo stack object

This object stores the current active state of the editor

Point 2: *Widgets are stateless and commutative*

Widgets have anti-hysteresis properties

When widgets are given a certain state, they behave
and render identically independent of the path taken

Colin Walters: “Immutable” ⟶ reprovisionable, anti-hysteresis
https://blog.verbum.org/2020/08/22/immutable-⟶-reprovisionable-anti-hysteresis/

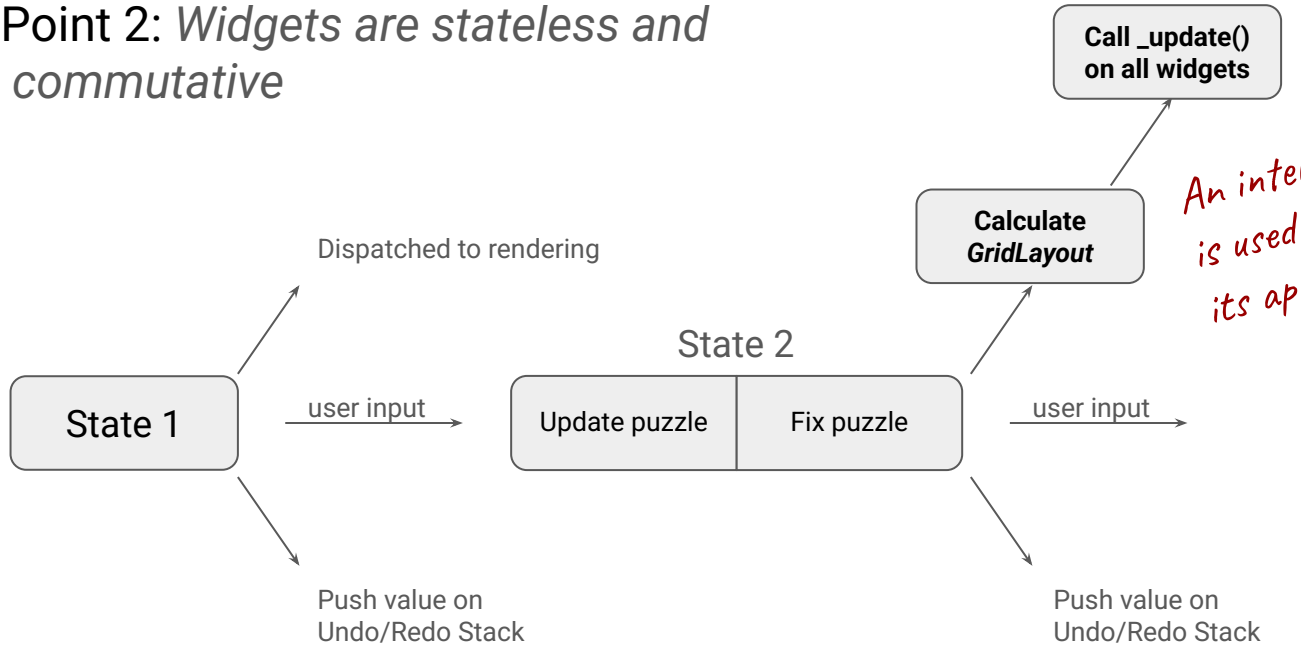
Point 2: *Widgets are stateless and commutative*

Writing *commutative* widgets:

1. Only one way of setting state:
   ```
   :update (value)
   ```
2. Only one way to get information out:
   ```
   ::changed (optional action_value)
   ```
3. Widgets cache just enough information to render
   a. They only redraw if information has changed in the ::update(value)
4. Widgets ***never*** change the state of the application
5. There are zero lateral signal connections, or signal connections to data
   a. (Hierarchical for propagation purposes in composite widgets is okay)

Anti-hysteresis style widgets are significantly easier to reason about

Point 2: *Widgets are stateless and commutative*

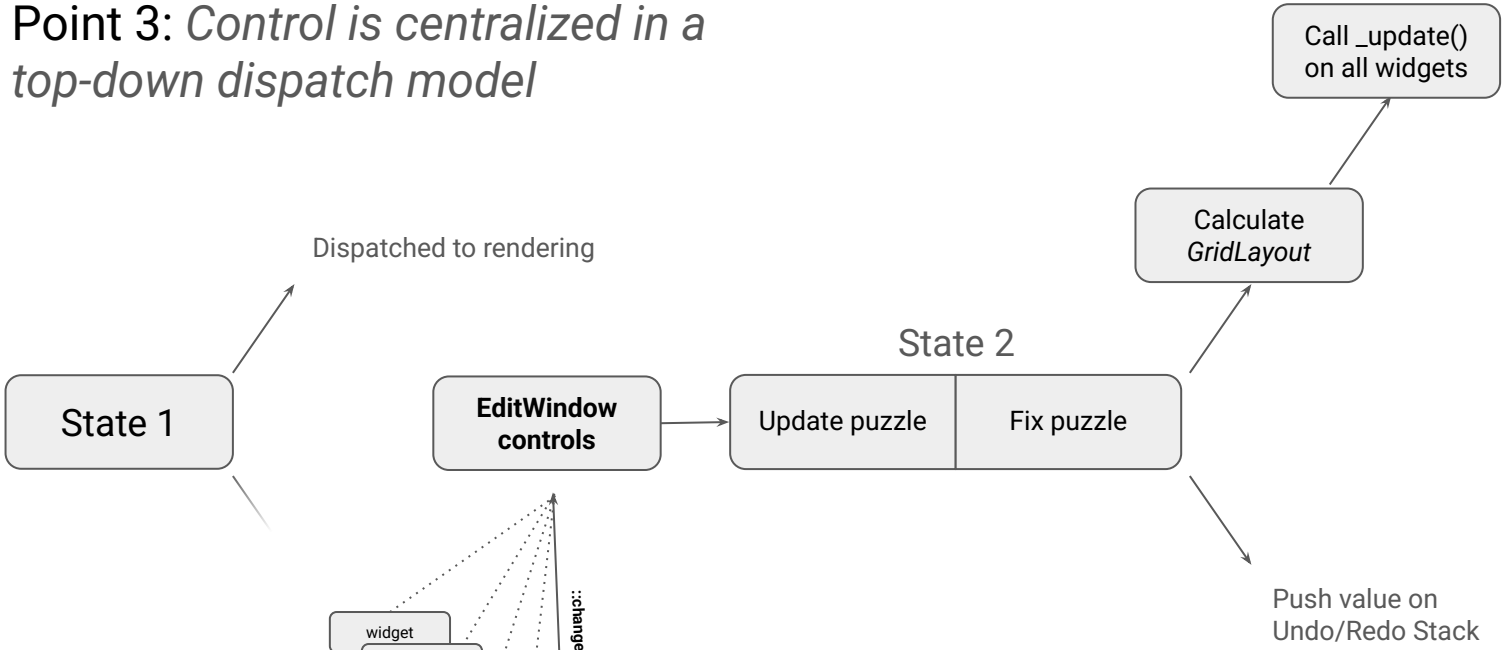


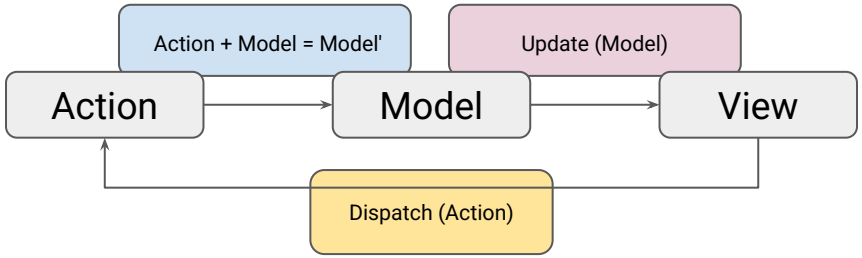
An intermediate grid representation is used to avoid each cell calculating its appearance from first principles

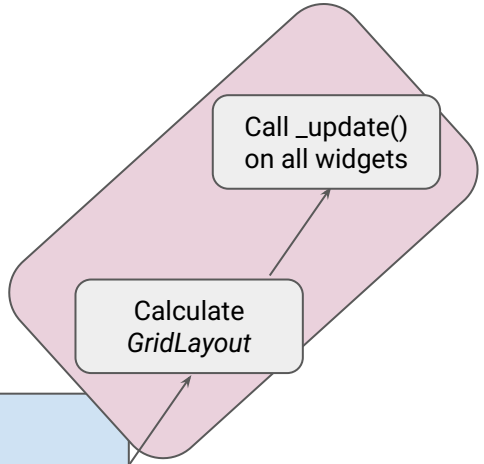
This is how undo/redo work!

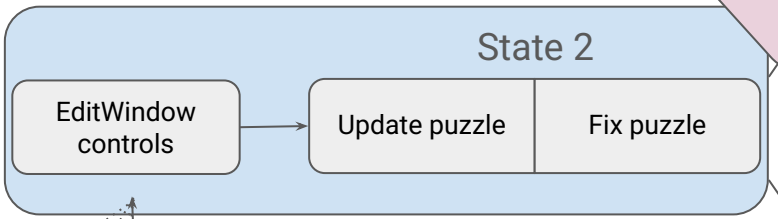
It's just another state to dispatch to the widgets

Point 3: *Control is centralized in a top-down dispatch model*



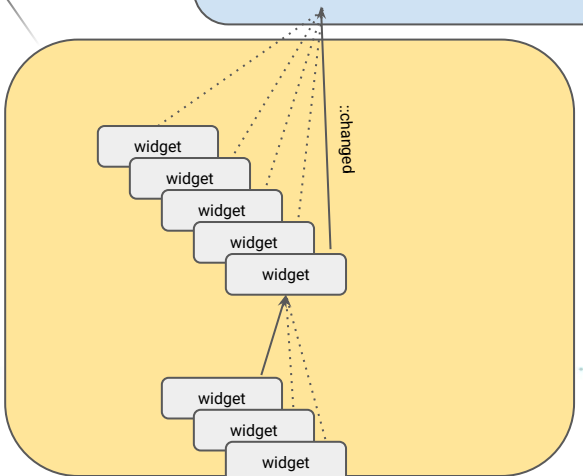






State 1

We can test across this barrier!



Push value on
Undo/Redo Stack

Point 4: *Complexity is concentrated*

Examples:

edit-window-controller.c (dispatch):
Translates between messy world of gtk and values

grid-state.c (crossword behavior):
Handles all mutations to values

puzzle-set-model.c (files):
Translates between messy world of file systems and puzzles

word-list.c (word lookup):
Finds potential word matches

Widgets and data structures are kept **as simple as possible**

Conclusions

- I'm a huge fan of this approach
  - It's the only reason we've been able to scale to this level of complexity
  - Protected us from toolkit changes: GTK3 → GTK4 → libadwaita → ???
  - Allowed new contributors to be able to orient themselves (relatively) quickly
  - Way fewer bugs than expected for an app this size

You should use it in your apps!

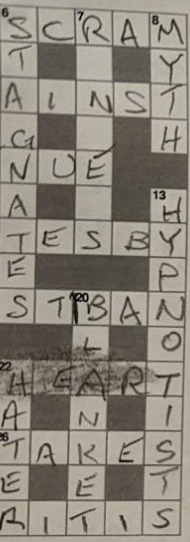
Questions?

https://gitlab.gnome.org/jrb/crosswords/

#crosswords on matrix

Or talk to me! <jrb@gnome.org>

5006 by David McLean



NAME

ADDRESS

POSTCODE

CONTACT NUMBER

EMAIL ADDRESS

A Cross Townsend micro-knurl fountain pen, worth £175, will be awarded to the first randomly selected correct solution. The next three win £125 Townsend micro-knurl ballpoint pens. To enter, complete the crossword and submit your entry. 18+ only. Closing date is 23:59 on the first Saturday after publication. Full T&Cs apply. Printed entries can be sent to: The Sunday Times Crossword 5006, PO Box 29, Colchester, Essex CO2 8GZ. For terms and conditions please visit http://bit.ly/2qSXOB9

DOWN

1 A Pole employed by one casting Hook? (7,3)
2 Pointers annoy grannies ultimately (7)
3 Agreeable blokes leaving southern state (5)
4 Senior on drugs charged along with royal (4-7)
5 Duck left out for afternoon munchies (3)
6 Does nothing get Satan annoyed with saint? (9)
7 Don't dash as much as one out on first delivery (7)

DOWN

1 Johnny heartlessly pinches hooter, showing a lot of cheek (5)
2 During Prohibition, wants whiskey in dark pen? (5,4)
3 So king admitted to self-importance (4)
4 Put off ruler, recoiling before something seedy (7)
5 Composer of verse in struggle with broadcaster (10)
6 Batting, keeping score up, one could do this to The Ashes (5)
7 Someone able to translate emigrant's novel (9)
8 Note ingredient of cheese gets thrown up (6)
9 For fun, one chills G&T, welcoming entertainer with precipitation (5,5)
14 Grade A is wasted on some students (4-5)
15 A better piano with jazz vocalist's singing style (1,8)
17 Keeper, Chelsea's No. 1, raised game before a right back seizes ball (7)
19 Pacific Island right by nice ground (6)
20 Slap for lacking manners (5)
22 Area surrounded by six sides seen by many people? (5)
24 Before kiss, take off top (4)
25 Before long, with light run (7)
26 Evenly slices possibly crude meat (6)

THE SUNDAY TIMES
Sunday, 08 May 2022