TU Graz

SCIENCE
PASSION
TECHNOLOGY

Gizual User Interface

Browser-Based Visualisation for Git Repositories

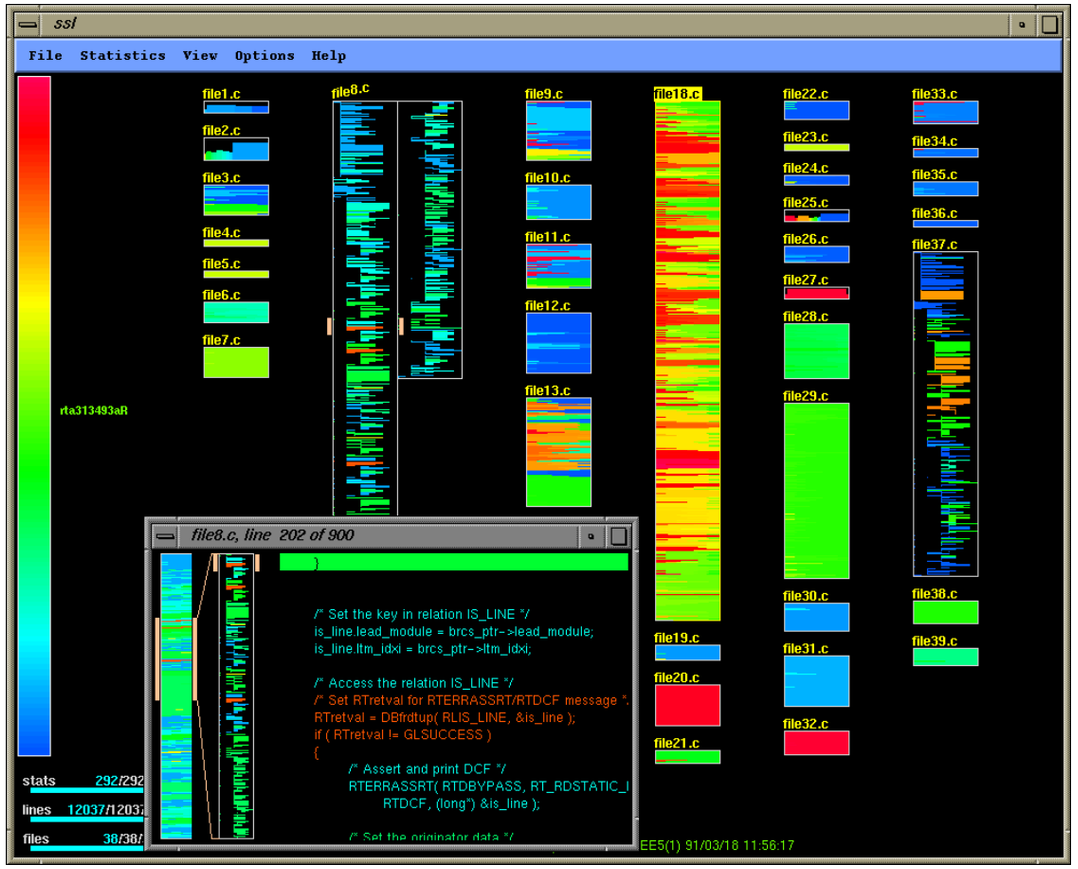
Master's Examination – Andreas Steinkellner

29 Jan 2025

gizual.com

Seesoft [1992]

- Visualisation tool for software repositories.
- Published by Eick et. al[1].
- Analyse up to 50,000 lines of code simultaneously.
- Interactive analysis.



Seesoft.
[Image extracted from Eick et al.[1] and used under §42f of Austrian copyright law.]

[1] Eick, Stephen G., Joseph L. Steffen and Eric E. Sumner Jr [1992]: *Seesoft: A Tool for Visualizing Line Oriented Software Statistics*; IEEE Transactions on Software Engineering 18.11 (Nov 1992) | doi: 10.1109/32.177365

Git

- Version control system.
- Free and open source[1].
- Industry standard.
- Powerful command line interface.
- Integrated in many native development tools.

[1] Git: https://git-scm.com/

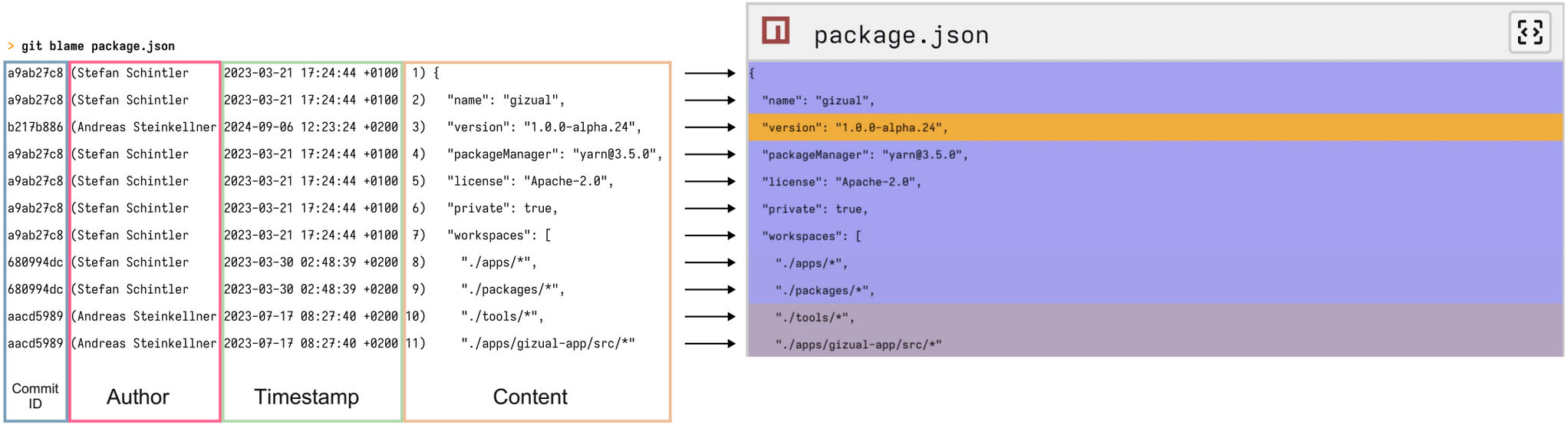
Git Blame

```
> git blame package.json
a9ab27c8 (Stefan Schintler     2023-03-21 17:24:44 +0100  1) {
a9ab27c8 (Stefan Schintler     2023-03-21 17:24:44 +0100  2)   "name": "gizual",
b217b886 (Andreas Steinkellner 2024-09-06 12:23:24 +0200  3)   "version": "1.0.0-alpha.24",
a9ab27c8 (Stefan Schintler     2023-03-21 17:24:44 +0100  4)   "packageManager": "yarn@3.5.0",
a9ab27c8 (Stefan Schintler     2023-03-21 17:24:44 +0100  5)   "license": "Apache-2.0",
a9ab27c8 (Stefan Schintler     2023-03-21 17:24:44 +0100  6)   "private": true,
a9ab27c8 (Stefan Schintler     2023-03-21 17:24:44 +0100  7)   "workspaces": [
680994dc (Stefan Schintler     2023-03-30 02:48:39 +0200  8)     "./apps/*",
680994dc (Stefan Schintler     2023-03-30 02:48:39 +0200  9)     "./packages/*",
aacd5989 (Andreas Steinkellner 2023-07-17 08:27:40 +0200 10)     "./tools/*",
aacd5989 (Andreas Steinkellner 2023-07-17 08:27:40 +0200 11)     "./apps/gizual-app/src/*"
```

Commit ID | Author | Timestamp | Content

Output from a Git blame command on the command line – sections annotated.
[Image created by the author of this presentation.]

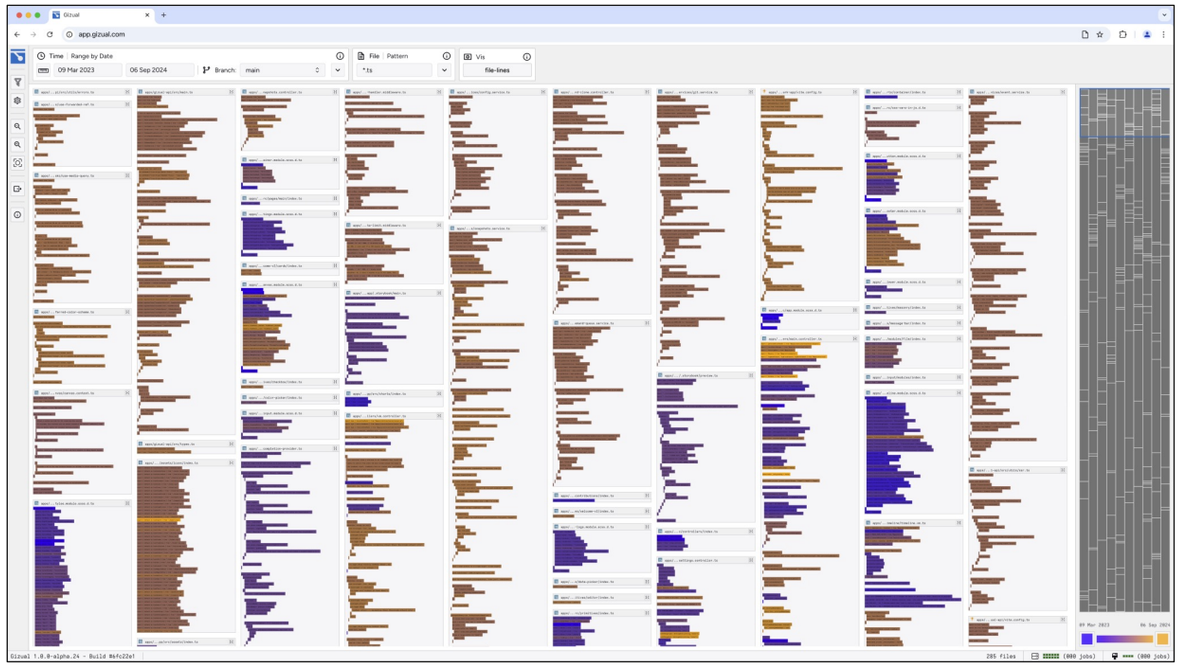
Git Blame vs. Gizual



Comparison of Git blame output (left) vs. Gizual (right, coloured by age of line of code).
[Image created by the author of this presentation.]

Gizual [2022 →]

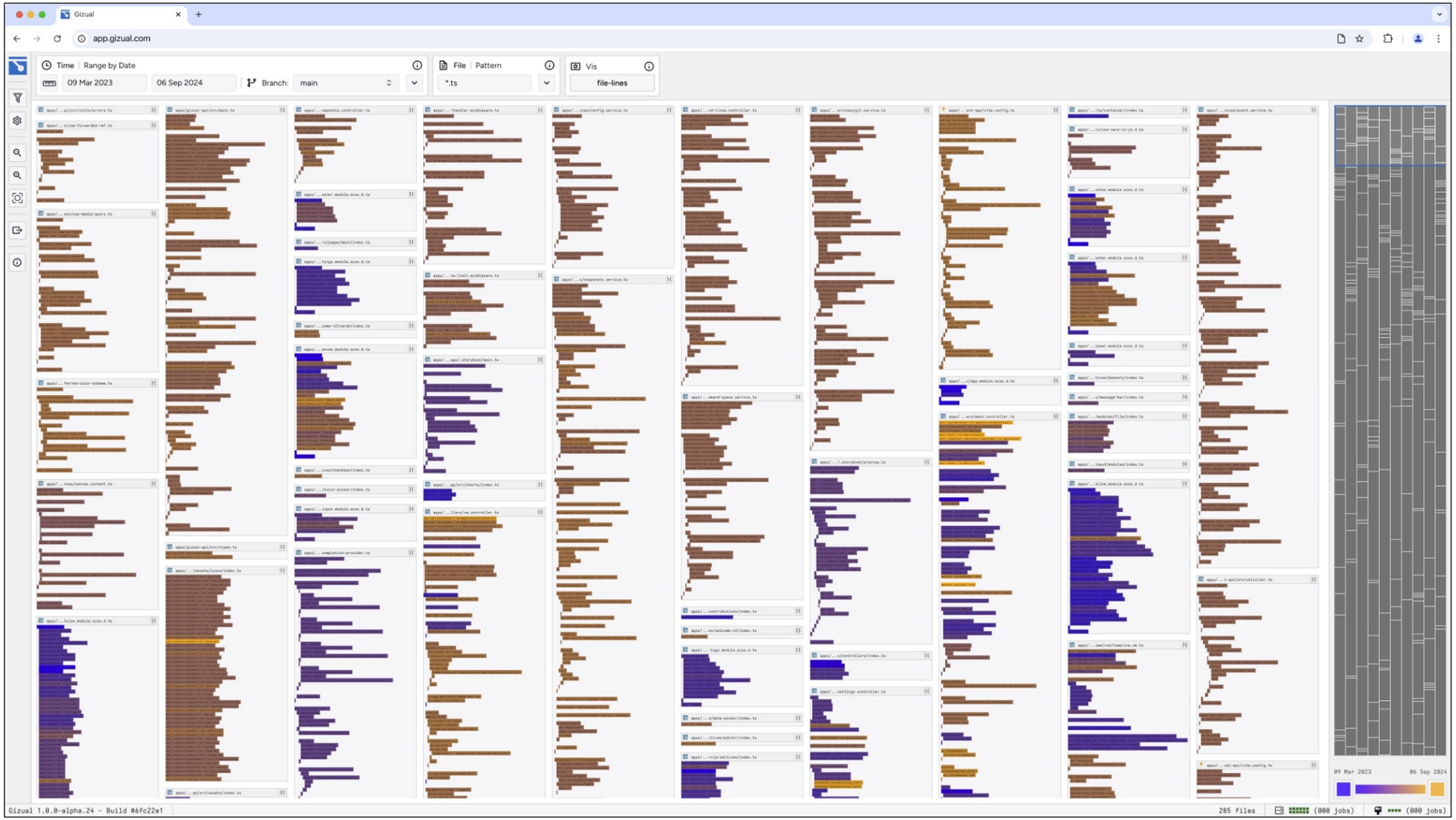
- Interactive visualisation tool for software repositories.
- Fast and easy to use web application[1].
- Local data processing.
- Custom queries for detailed analysis.
- Stack: React, TypeScript, MobX, Mantine, Libgit2, Rust.



Gizual: Main interface.
[Image created by the author of this presentation.]

[1] Gizual: https://gizual.com

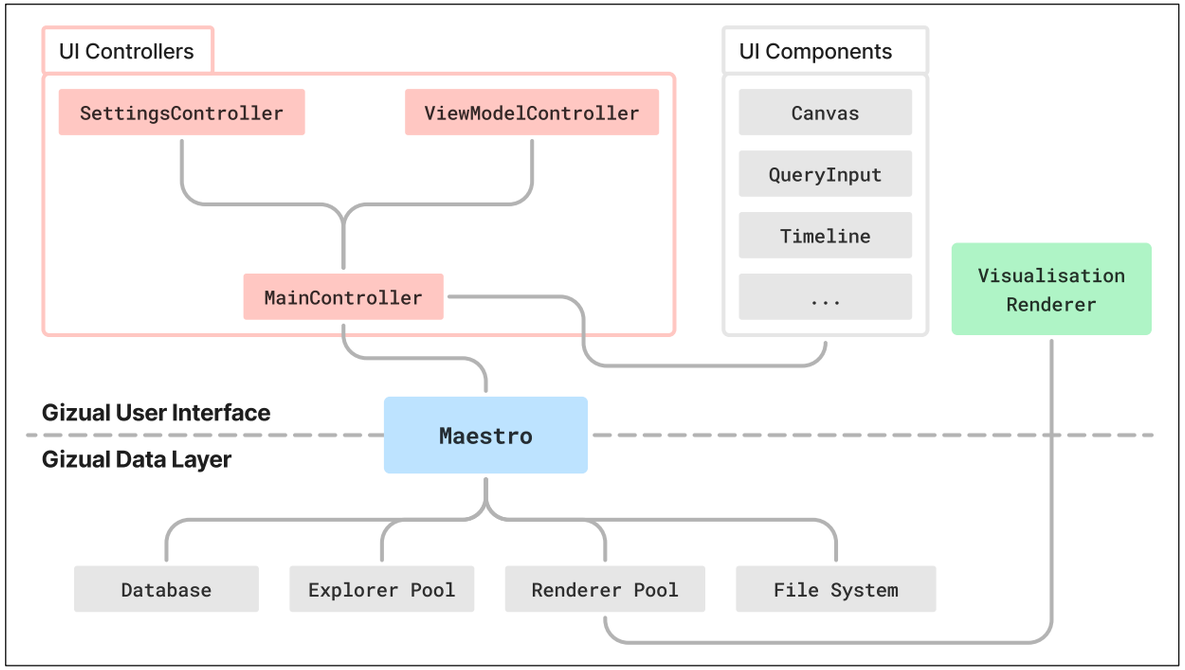
Gizual: Live Demo



Architecture

- *This work:* User interface, visualisation, interactivity, user experience and accessibility.

- *Data Layer (Stefan Schintler)*[1]: Git exploration, Web Worker parallelisation, file management.



Gizual architecture and project split.
[Image created by the author of this presentation.]

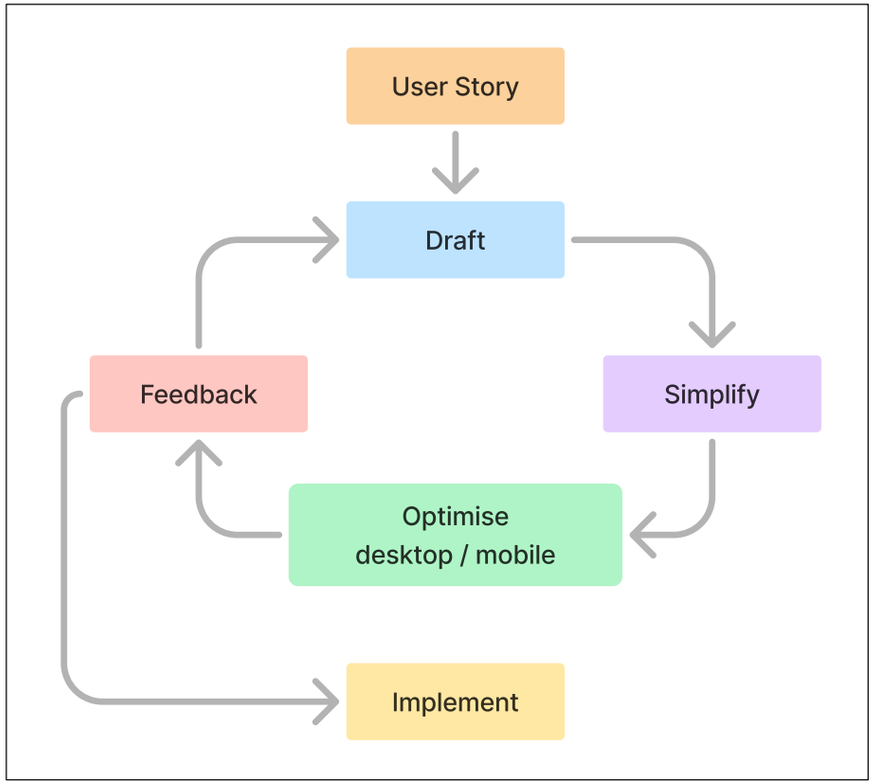
[1] Schintler, Stefan [2024]: *Gizual Data Layer: Enabling Browser-Based Exploration of Git Repositories* | url: https://ftp.isds.tugraz.at/pub/theses/sschintler-2024-msc.pdf

User Experience Goals

- Ease of use and simplicity.
- High performance.
- Minimise required interactions to desired output.
- Guide user focus on main content.
- Multiple input methods.
- Accessible where possible.
- Optimised for desktop and mobile.
- Export functionality.

UI Components: Thought Process

- Start with user story.
- Draft design for possible user interface.
- Simplify the concept.
- Consider narrow viewports and touch devices.
- Test on desktop and mobile.
- Gather feedback and iterate.



User interface design process.
[Image created by the author of this presentation.]

Welcome Screen

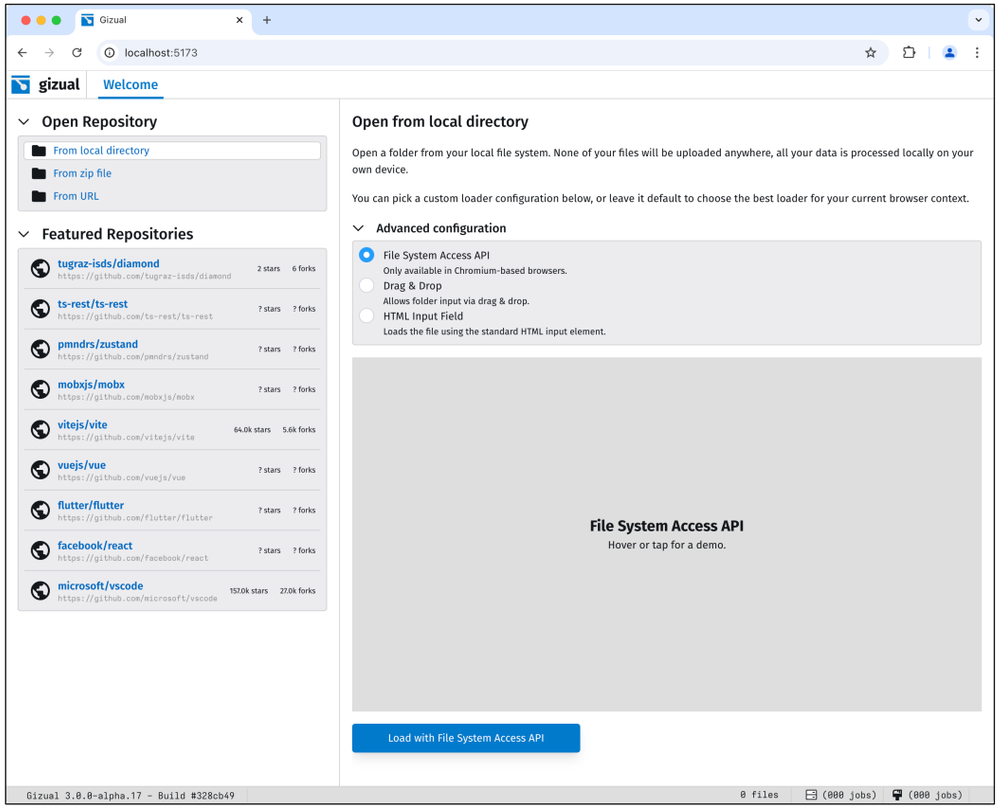
Why?

- Support for different browsers and file loaders.
- Performance differences between loaders.
- Address privacy concerns of users.
- Provide users with a choice.

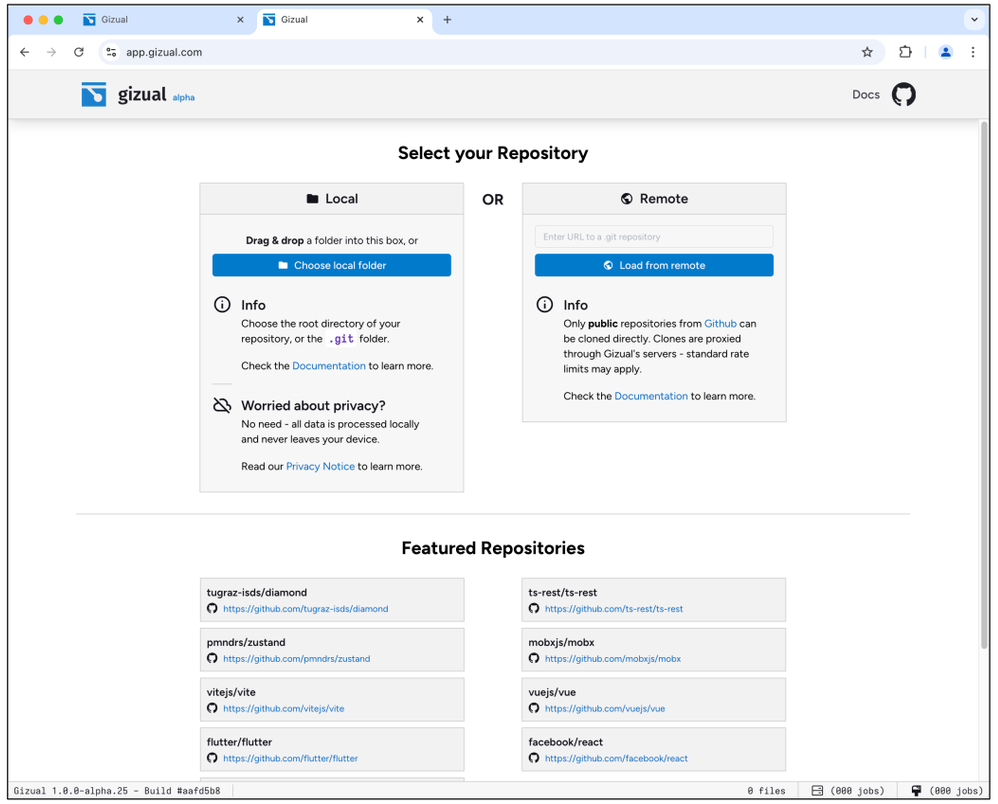
How?

- Separate screen before main application.
- Multiple designs and user flows were tested.

Welcome Screen: Iterative UX Improvements

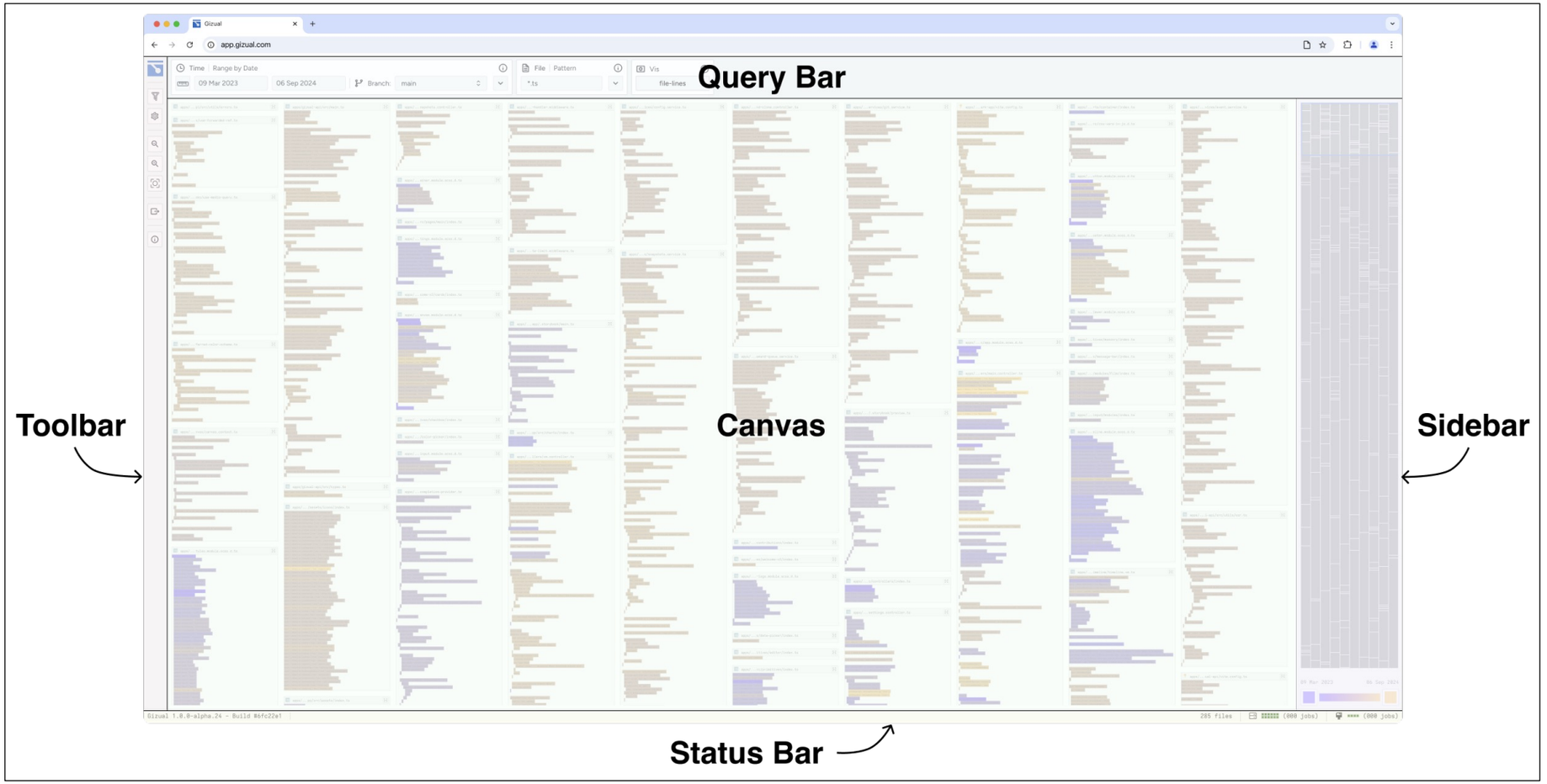


Gizual's welcome screen [Mar 2024].
[Image created by the author of this presentation.]



Gizual's welcome screen [current].
[Image created by the author of this presentation.]

User Interface Details

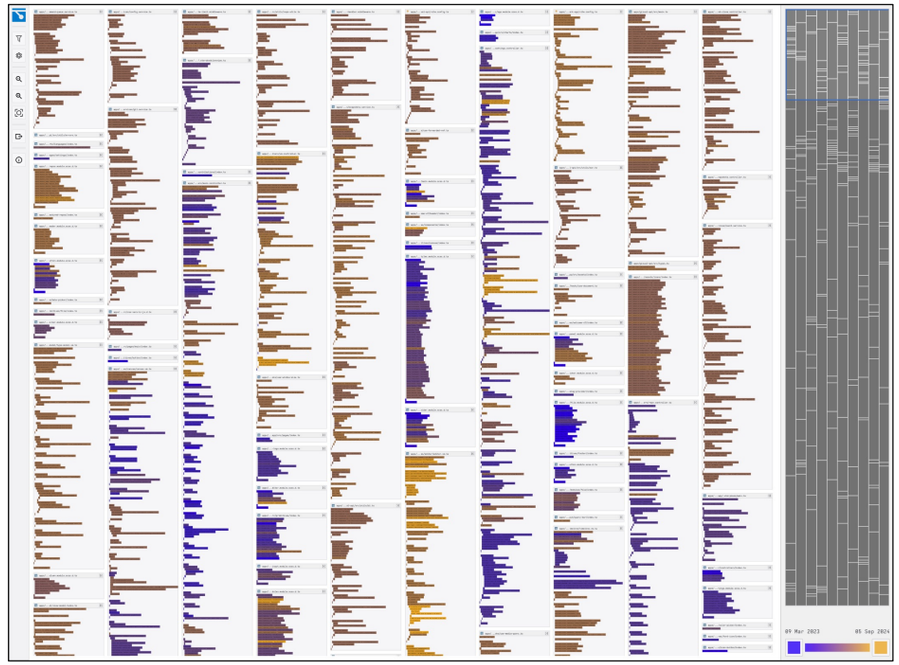


Gizual's main user interface with annotated regions.
[Image created by the author of this presentation.]

Canvas

📋 Requirements:

- Interactive, with high performance.
- Display hundreds (thousands) of files.
- Stable layout.
- Easily navigable.
- Good user experience on all viewport sizes and input devices.



Gizual's Canvas component.
[Image created by the author of this presentation.]

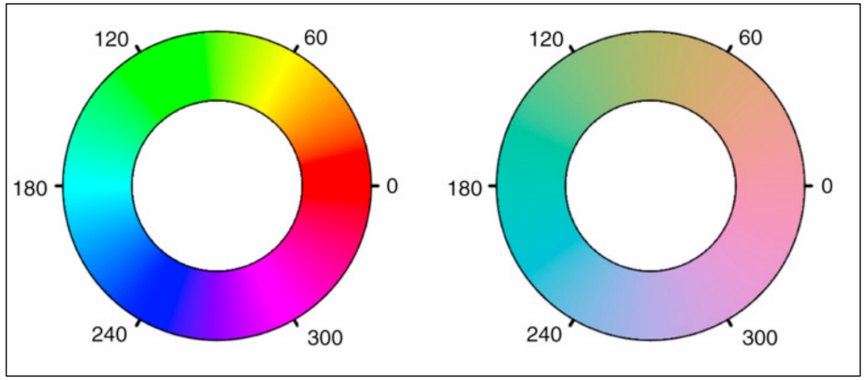
Canvas

💡 **Approach:**

- Each file rendered into a single image (block).
- Entire Canvas transformed on zoom, pan or pinch → CSS for high performance.
- Positioning of images in masonry layout.
  - Not available with pure CSS → Custom JavaScript implementation.
- Custom Minimap and Legend components support orientation.

Visual Encoding (1)

- Colour lines of code based on age or author.
- Age encoding: Gradient with two user-defined values.
- Author encoding: HCL palette with custom user overrides.
  - RGB colour space not uniformly distributed → use HCL colour space.
  - Custom algorithm based on d3-color[1] for stable results.



Comparison of HSV-based (left) and HCL-based (right) colour wheel.
[Image extracted from Zeileis et al.[2] and used under §42f of Austrian copyright law.]

[1] d3-color: https://d3js.org/d3-color
[2] Zeileis, Achim and Hornik, Kurt and Murrell, Paul [2009]: Escaping RGBland: Selecting Colors for Statistical Graphics; Computational Statistics & Data Analysis 53.9 (Jul 2009) | doi: 10.1016/j.csda.2008.11.033

Visual Encoding (2)

Model	Parameters	Details
RGB	Red, Green, Blue	Non-uniform, device dependent.
HSL	Hue, Saturation, Lightness	Perceptually-oriented, based on RGB.
HSV	Hue, Saturation, Value	Used interchangeably with HSL.
CIE XYZ	Tristimulus response (eye)	Perceptually uniform reference model, based on human vision experiments.
CIE Lab	Lightness, Green-to-red, Blue-to-yellow	Perceptually uniform, device independent.
HCL	Hue, Chroma, Luminance	Perceptually uniform, based on CIE.

Comparison of commonly used colour spaces.
[Table created by the author of this presentation, based on data from Noor et. al.[1]]

HCL colour band, created in Gizual for a set of ten authors.
[Image created by the author of this presentation.]

[1] Ibraheem, Noor & Hasan, Mokhtar & Khan, Rafiqul Zaman & Mishra, Pramod [2012]: Understanding Color Models: A Review; ARPN Journal of Science and Technology
url: https://www.researchgate.net/publication/266462481_Understanding_Color_Models_A_Review

Rendering

📋 Requirements:

- Parallelised rendering.
- Distributed workload across web workers.
- Reusable for different visualisations and outputs.

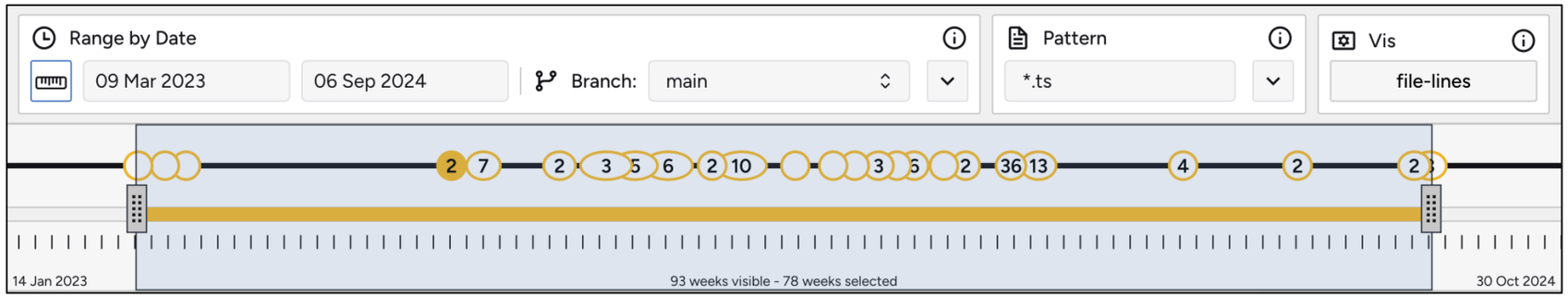
💡 Approach:

- Rendering tasks for each block (file).
- Generalised rendering interface.
- Class-based file and module structure.

Query Bar (1)

📋 Requirements:

- Everything accessible within few clicks.
- Modular setup, easily extendible.
- Consistent look and feel.

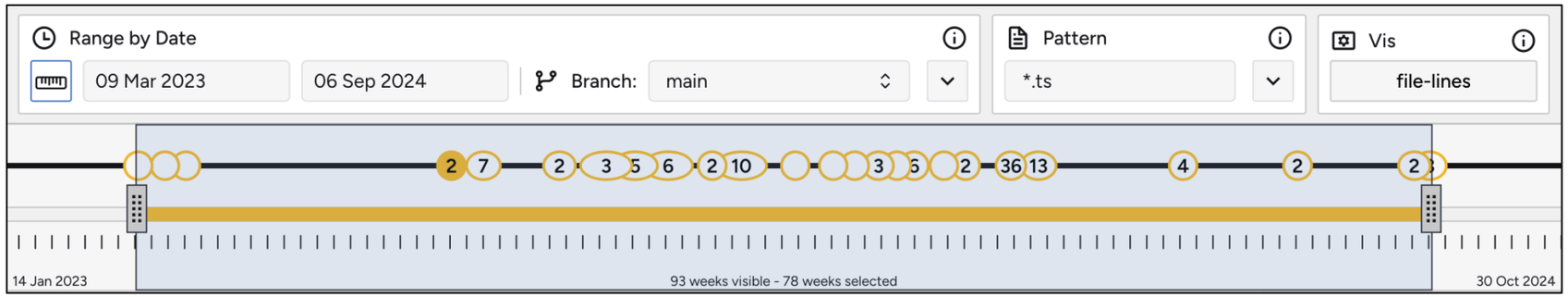


Gizual's Query Bar, with the Timeline component expanded.
[Image created by the author of this presentation.]

Query Bar (2)

💡 Approach:

- Generalised base module exposes common functionality.
- Individual modules grouped by category.
- Modules directly attached to query interface via Maestro.

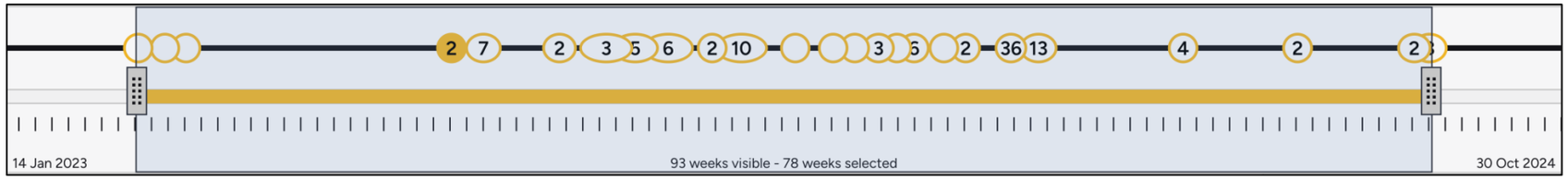


Gizual's Query Bar, with the Timeline component expanded.
[Image created by the author of this presentation.]

Query Bar: Timeline (1)

📋 Requirements:

- Easy selection of time range.
- Visual display of commits.
- High performance.

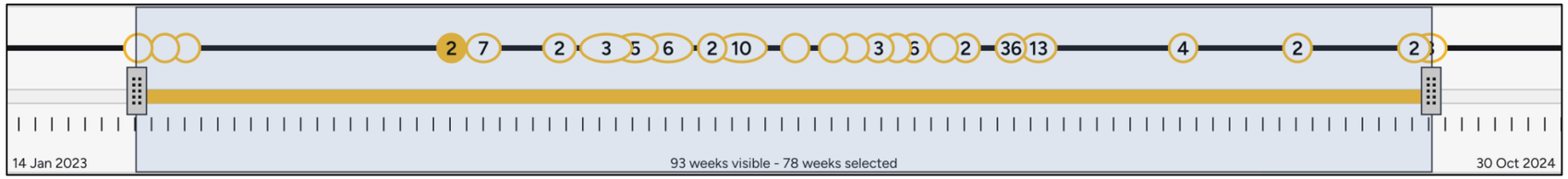


Gizual’s Timeline component.
[Image created by the author of this presentation.]

Query Bar: Timeline (2)

💡 Approach:

- SVG-based rendering with interactive HTML on top.
- Interactivity for zooming, panning and dragging (mouse and touch).
- Viewport pre-rendered to improve interactivity performance.



Gizual's Timeline component.
[Image created by the author of this presentation.]

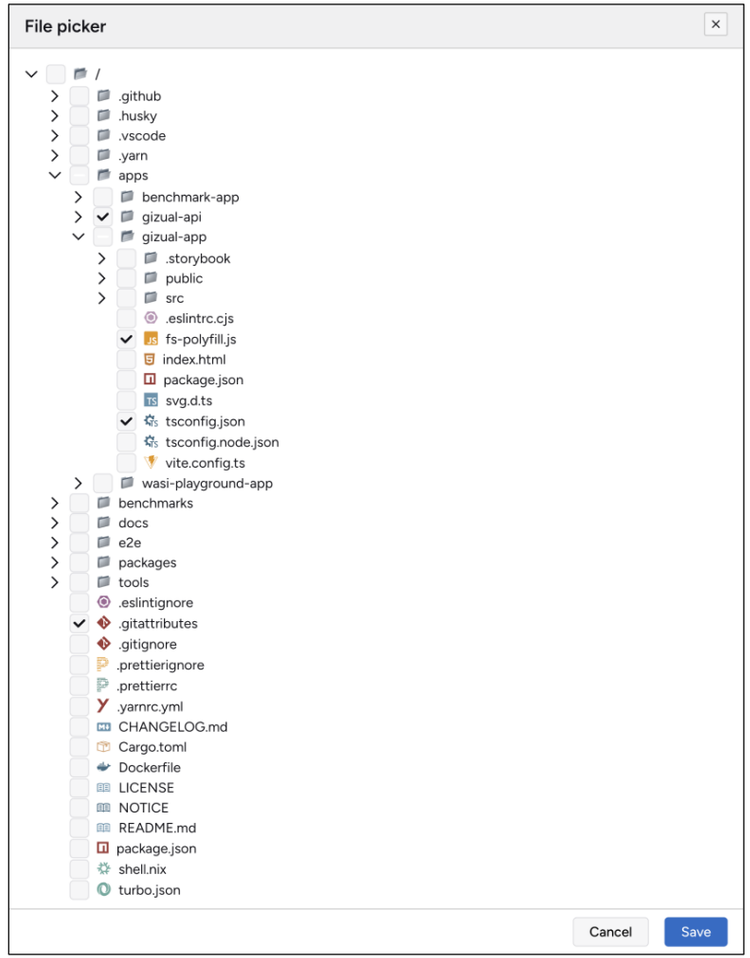
Query Bar: File Tree

📋 Requirements:

- Support for thousands of files.
- Partial selection for folders.
- File type icons.

💡 Approach:

- Limited recursive pre-rendering (3 levels).
- Internal tree representation in flat structure.



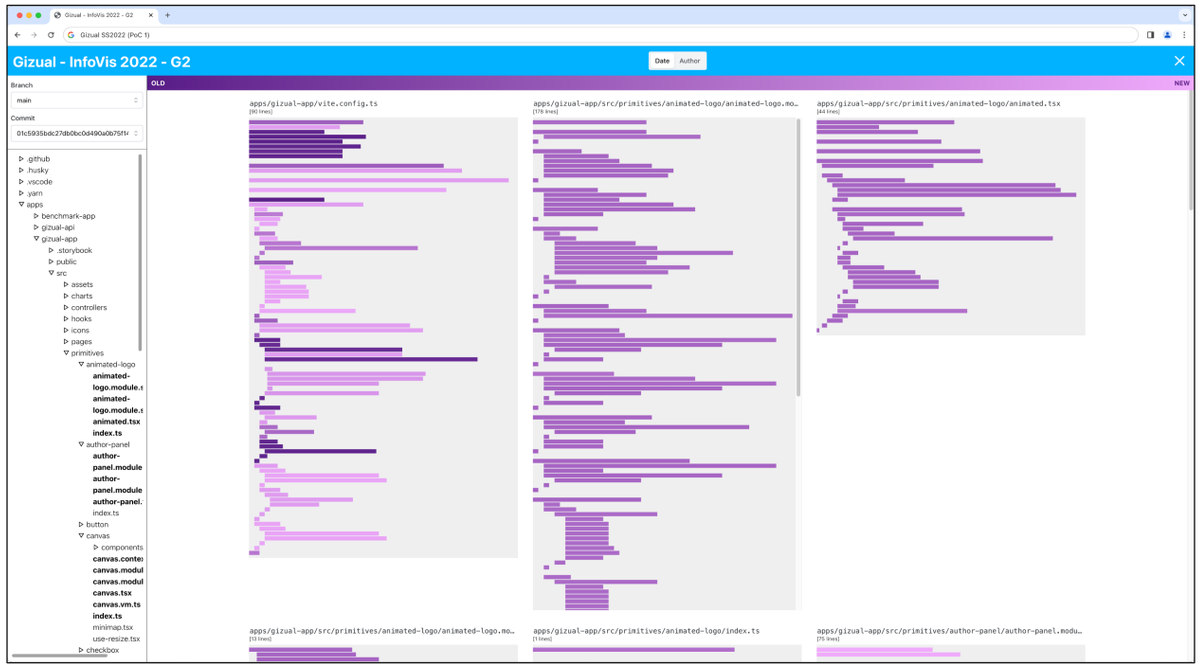
Gizual‘s File Tree component.
[Image created by the author of this presentation.]

Thank You!

gizual.com

Gizual Proof of Concept 1 [SS 2022]

- Limited interaction and metrics.
- Frequent UI freezes.
- Slow canvas performance.
- No customisation of results.

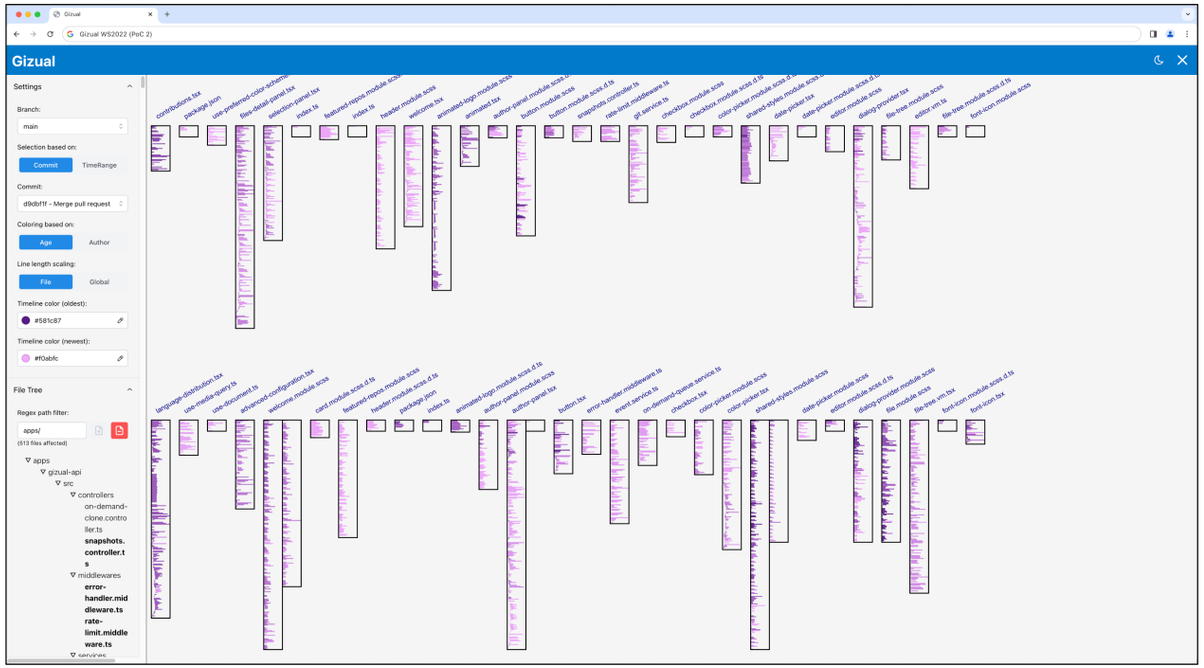


POC1[1]: [May 2022 → Jun 2022]
[Image created by Korduba, Schintler and Steinkellner[1].]

[1] Gizual - Repository Visualization for Git (SS 2022), seminar paper: https://gizual.com/resources/gizual-paper-ss2022.pdf

Gizual Proof of Concept 2 [WS 2022/23]

- Max. 50 files.
- Limited support for Firefox.
- Inefficient blame performance.



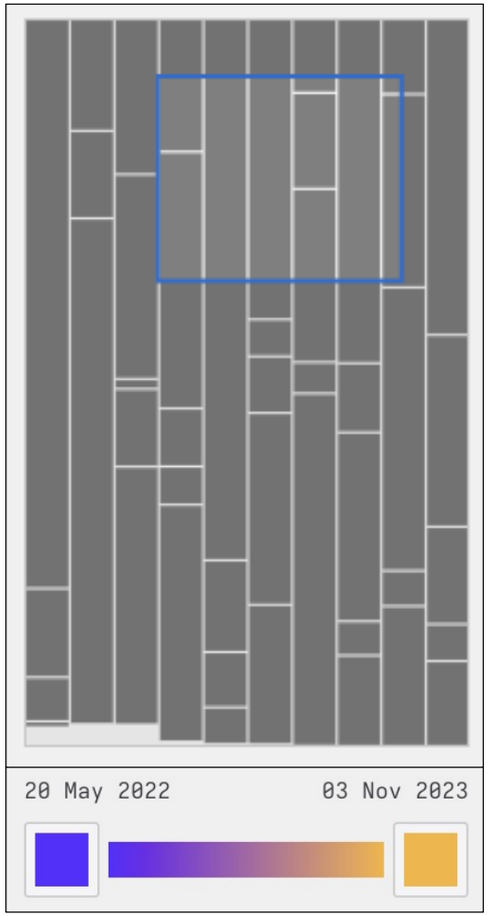
POC2[1]: [Dec 2022 → Jan 2022]

[Image created by Pinheiro de Souza, Schintler and Steinkellner[1].]

[1] Gizual - Repository Visualization for Git (WS 2022), seminar paper: https://gizual.com/resources/gizual-paper-ws2022.pdf

Canvas: Details

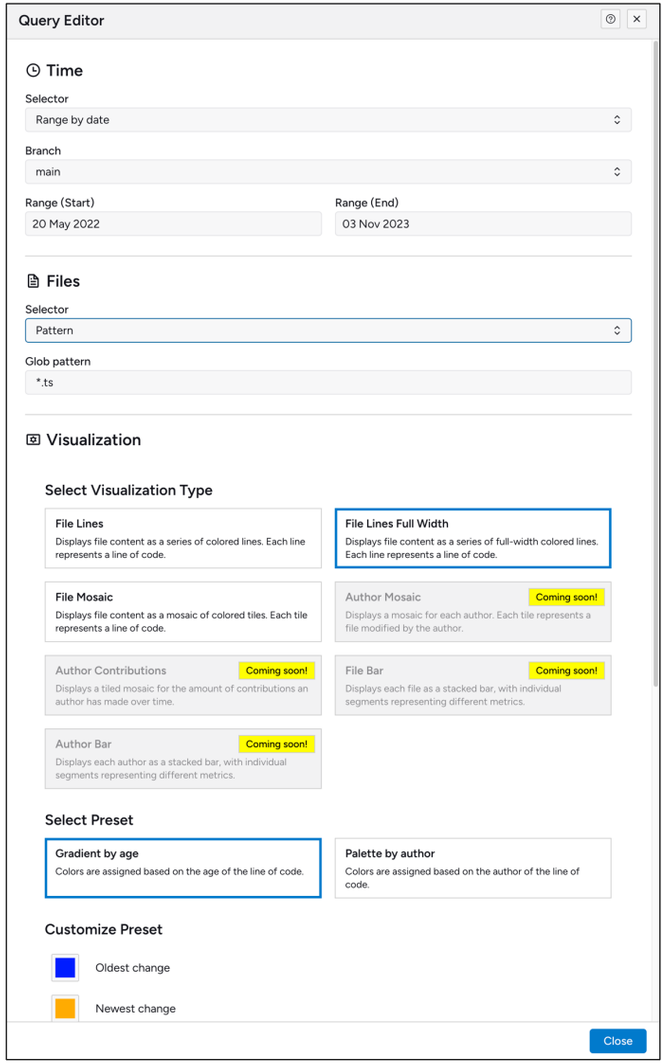
- Intuitive interactivity:
  - Zoom
  - Pan
  - Pinch
- Custom minimap implementation for overview in large visualisations.
- Custom legend component.
  - Quick access to gradient colours.
  - Show selected time range.



Gizual's Minimap and Legend components.
[Image created by the author of this presentation.]

Query Editor

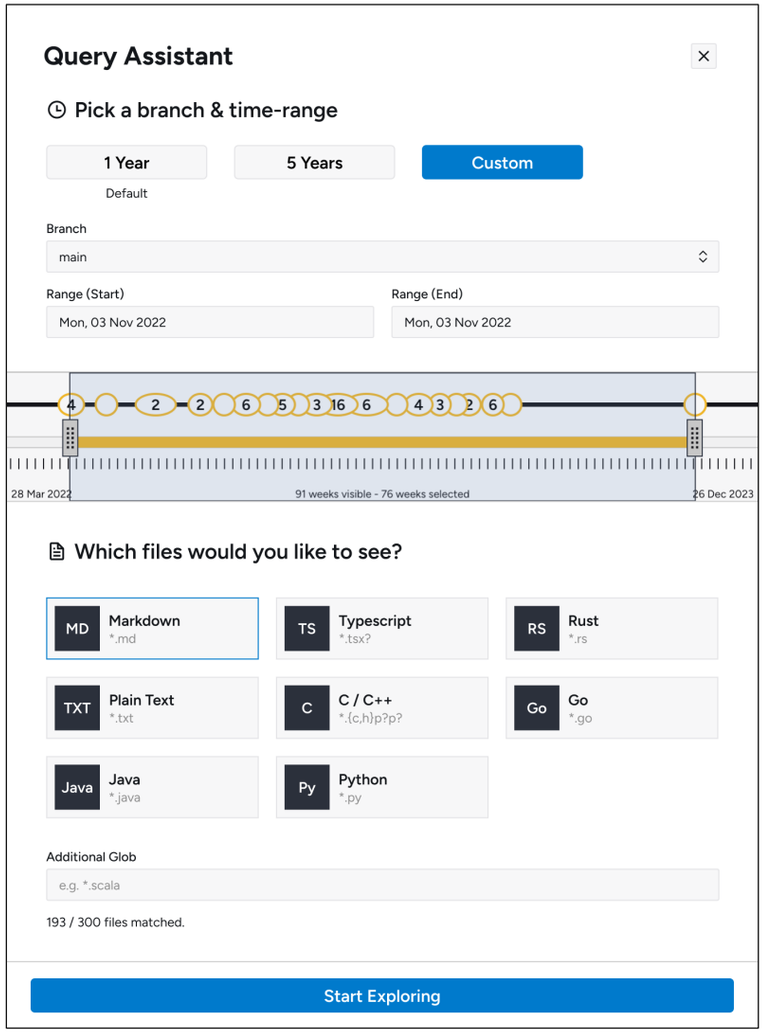
- Replaces Query Bar on devices with narrow viewports.
- Provides a simplified selection of modules.



Gizual's Query Editor.
[Image created by the author of this presentation.]

Query Assistant (Draft)

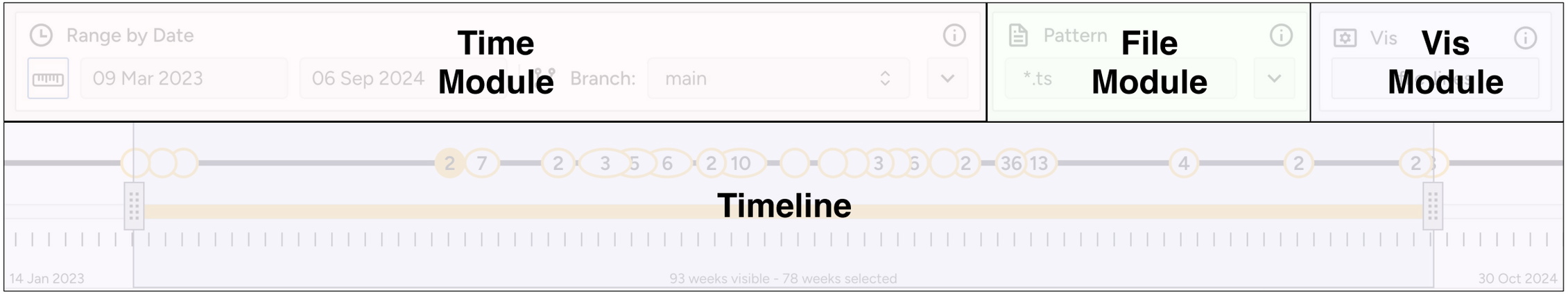
- Easier way for new users to get started.
- Contains a subset of available options.
- Highly focused on selecting a range and files to start immediately.



Gizual's Query Assistant.
[Image created by the author of this presentation.]

Query Bar Modules

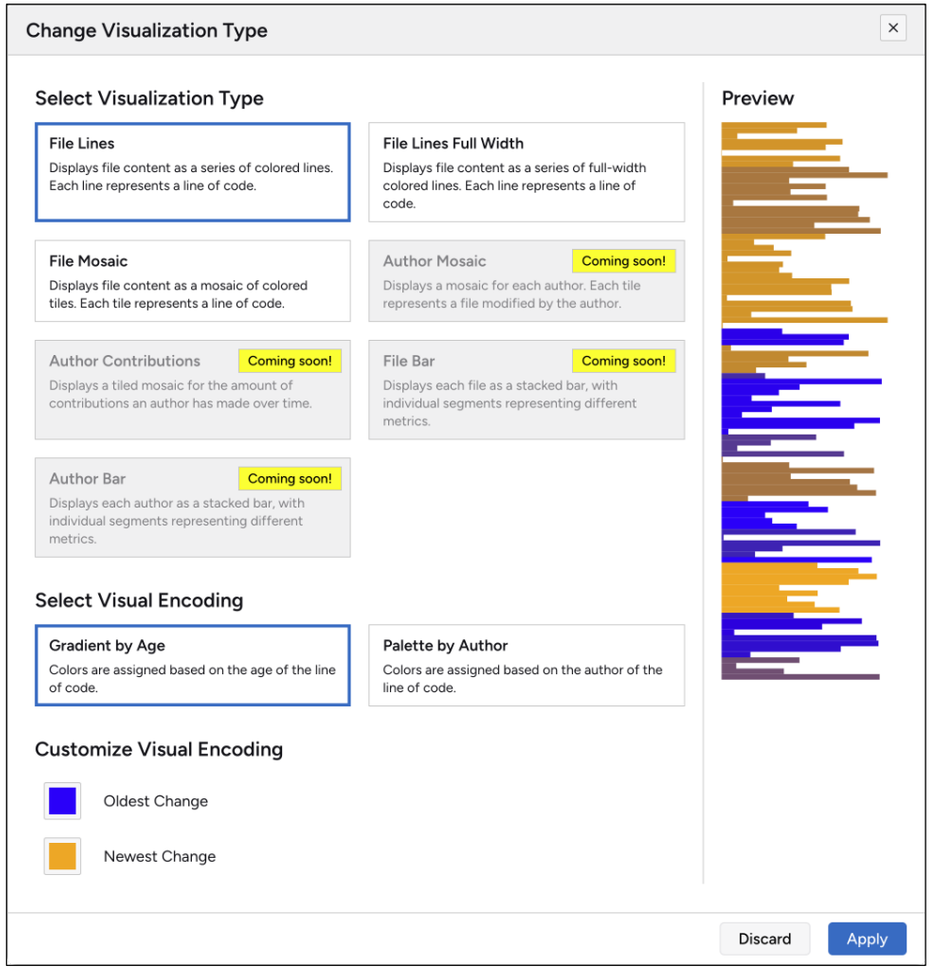
- Module groups provide interchangeable modules.
- Visual layout always consistent.
- Modules can be swapped with one button.



Gizual's Query Bar, annotated with module information.
[Image created by the author of this presentation.]

Visualisation Type Dialogue

- Deliberate use of dialogue instead of direct display in Query Bar.
- Overview of visualisation types.
- Choice between visual encodings.
- Preview of selected visualisation type in selected encoding with customised colours.



Gizual's Visualisation Type Dialogue.
[Image created by the author of this presentation.]

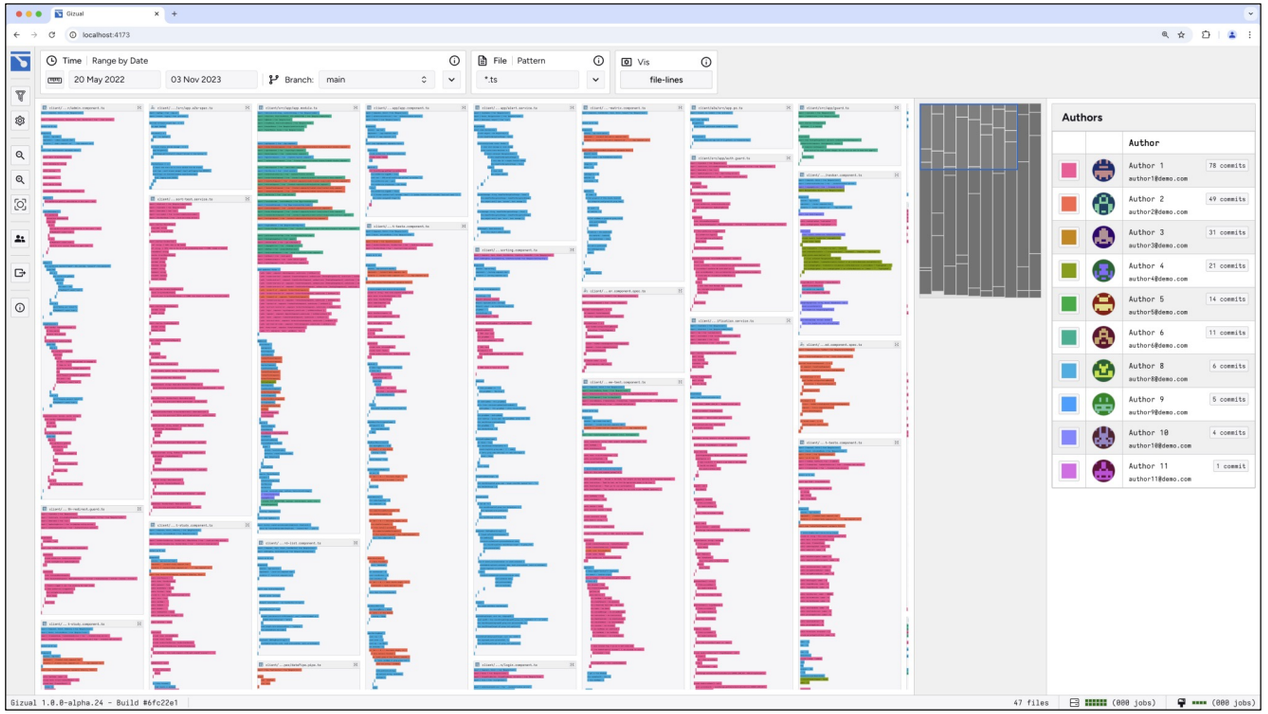
Mosaic Mode

- Alternative visualisation mode.
- 10 lines of source code grouped into a single line of tiles.
- Condensed representation.
- No rendered text content.

Gizual’s line mode (left) compared to mosaic mode (right).
[Image created by the author of this presentation.]

Palette by Author Encoding

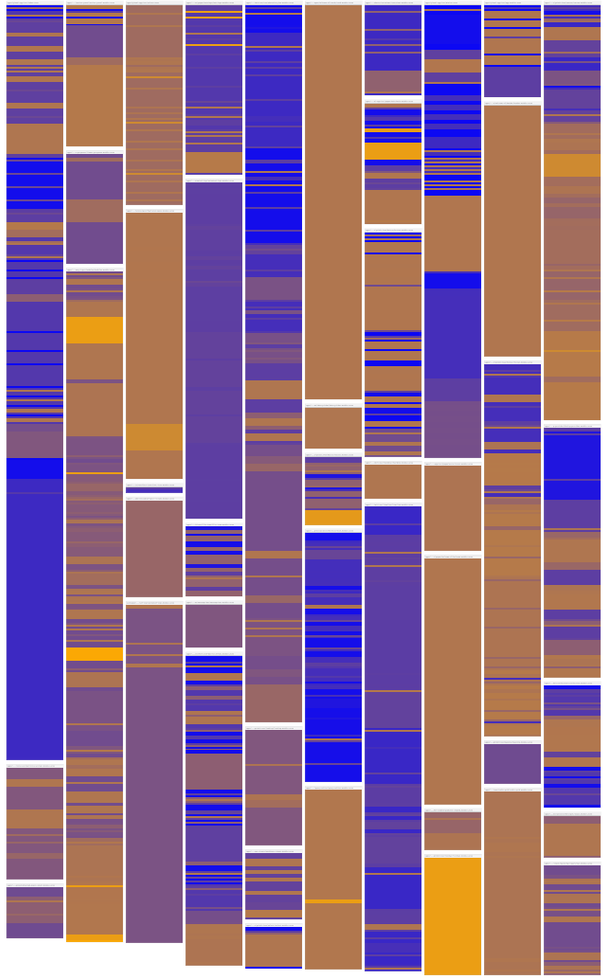
- Colour based on author of line of code instead of timestamp.
- HCL colour space for equal distribution per default.
- Author panel automatically visible.
- Colours can be customised by user.



Gizual's main interface with a visualisation in Palette by Author visual encoding.
[Image created by the author of this presentation.]

SVG Export

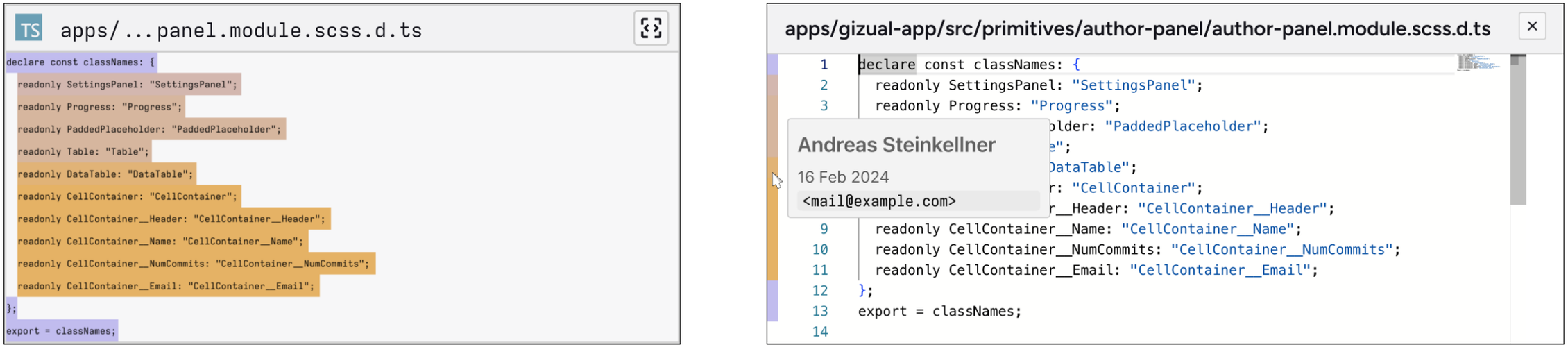
- Stable masonry algorithm.
- Files assigned into columns and sorted based on rendered height.
- File header rendered without interactive elements.



Gizual's SVG Export.
[Image created by the author of this presentation.]

Integrated Code Editor

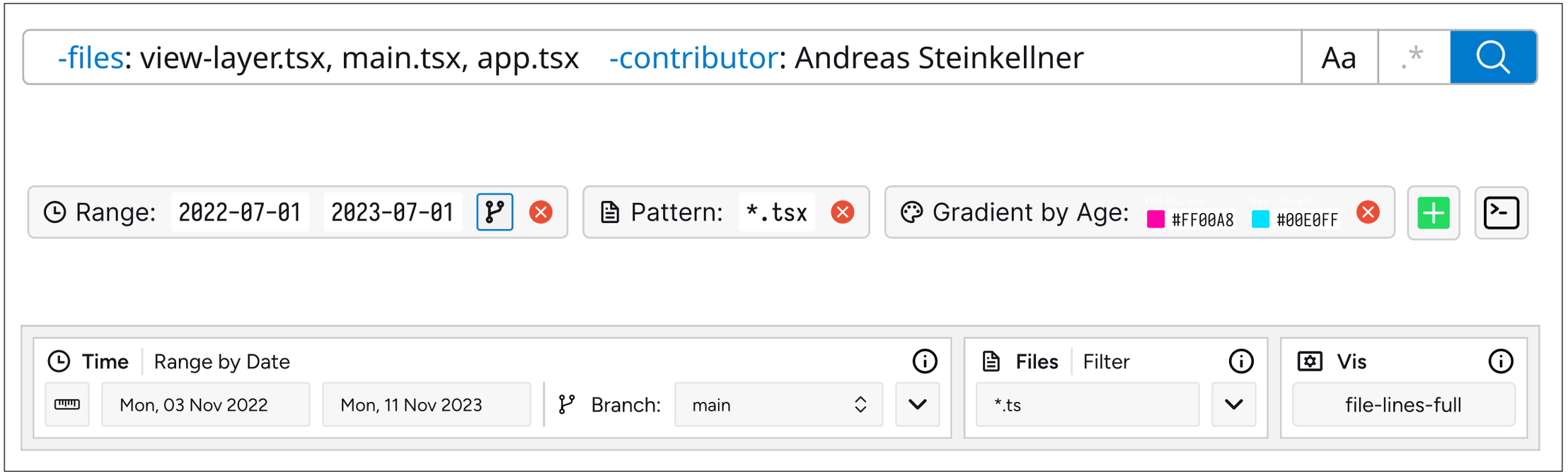
- Based on Monaco[5] code editor.
- Custom line gutter and tooltip.



Gizual file content in visualisation (left) vs. file content in integrated editor (right).
[Image created by the author of this presentation.]

[5] Monaco: https://microsoft.github.io/monaco-editor/

Query Bar Iterations



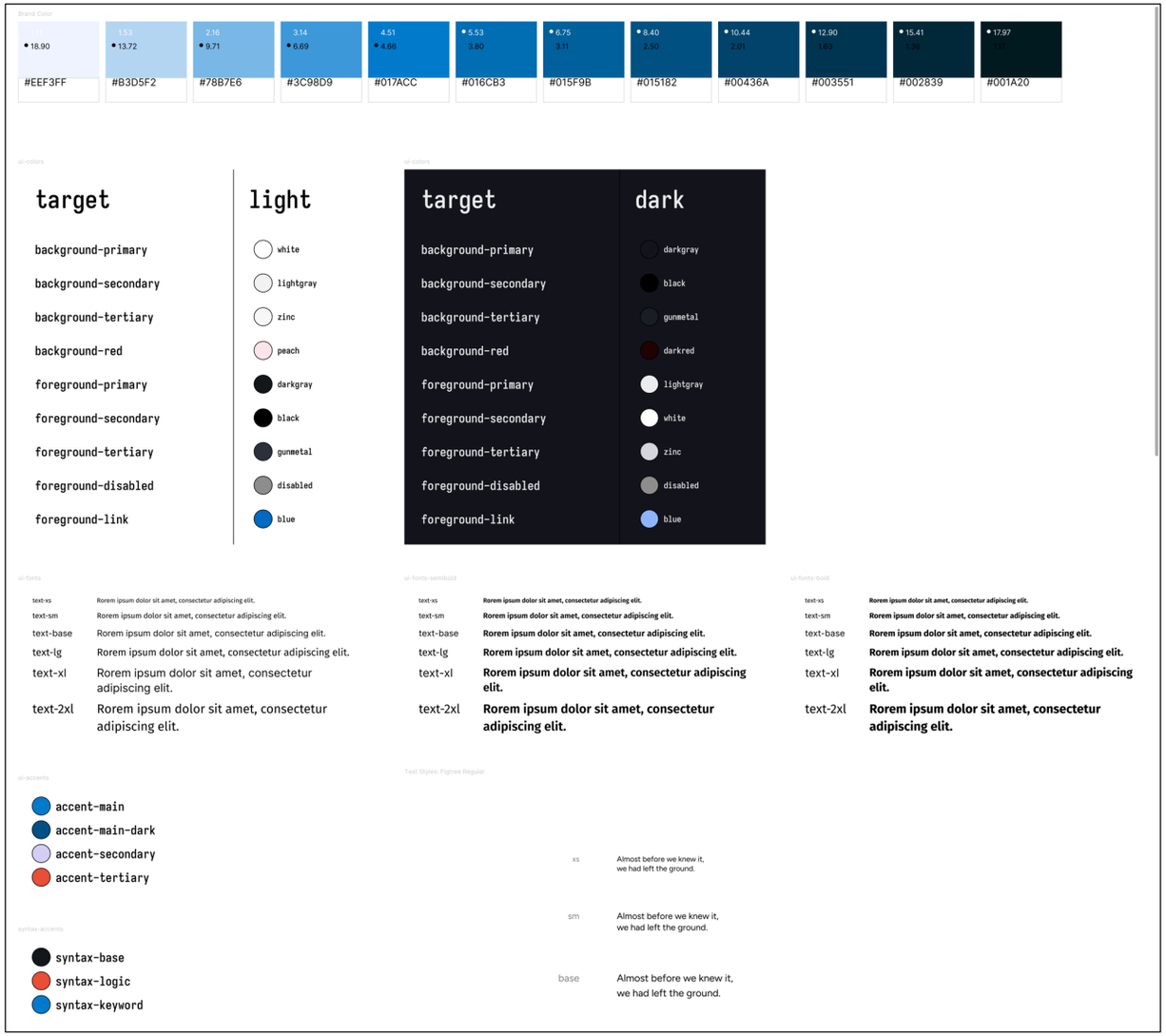
Different iterations of Gizual's Query Bar, ordered top to bottom from oldest to current.
[Image created by the author of this presentation.]

Development Toolchain

- React + TypeScript.
- State management: MobX.
- Component library: Mantine.
- Design: Figma.
- Styling: SCSS.

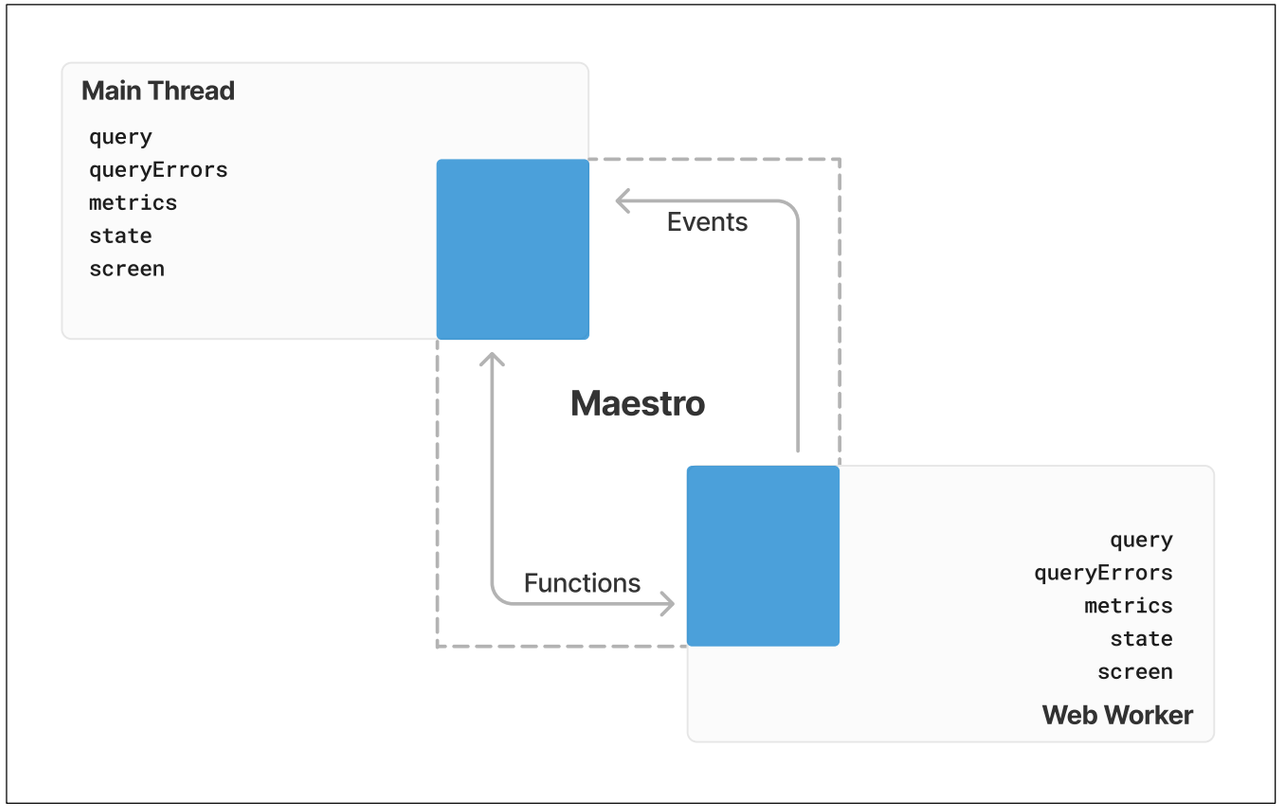
Figma

- Heavily used for design and prototyping.
- Consistent look and feel across separated UI components.

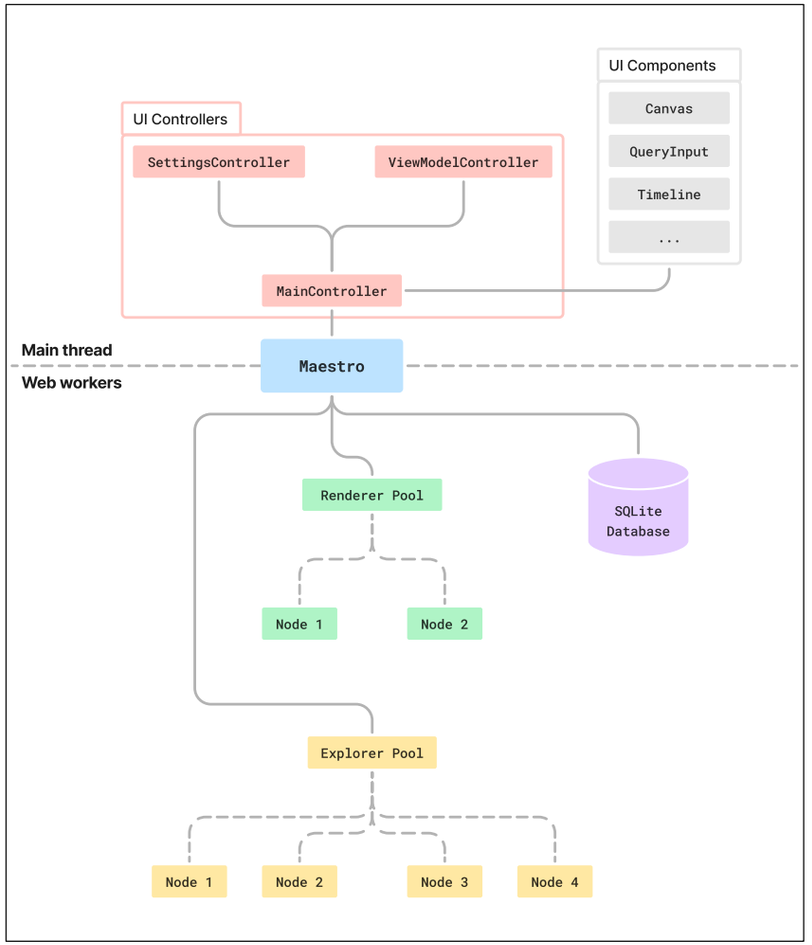


Gizual theme in Figma.
[Image created by the author of this presentation.]

Architecture & Maestro



Maestro controller.
[Image jointly created by the author of this presentation and Stefan Schintler[1].]



Gizual architecture.
[Image jointly created by the author of this presentation and Stefan Schintler[1].]

[1] Schintler, Stefan [2024]: *Gizual Data Layer: Enabling Browser-Based Exploration of Git Repositories* | url: https://ftp.isds.tugraz.at/pub/theses/sschintler-2024-msc.pdf

User Interface Structure

- `<component.module.scss>`
  SCSS module with all required styles for the component.
- `<component.module.scss.d.ts>`
  Automatically generated types for the SCSS classes.
- `<component.tsx>`
  Main entry file for the component.
- `index.ts`
  Clean export for entry file.

Future Work (1)

- Canvas performance improvements through SVG-based rendering.
- Custom analytics with charts.
- Timeline performance improvements through CSS transforms.
- Binning for visual encoding.
- Brushing for specific lines of code.
- Hierarchical structure overview (treemap).
- SVG export with file content.
- Native build.

Future Work (2)

- Quick toggles: Visual encoding.

Ideas:

- Text diff between revisions.
- Merge conflict detector.
- Filter by commit message.