Visual programming and data analysis

Prof.Dr. Darian M. ONCHIS

A visual programming language, called in English "VPL" (Visual Programming Language), is a programming language that allows a user to program a computer in a two-dimensional or multi-dimensional way.

Examples of such additional dimensions are: the use of spatial relations, the use of multi-dimensional objects, the use of the time dimension to specify "before-after" semantic relations.

Each multi-dimensional tuple or relation is a token. A collection of one or more tokens is a visual expression.

Visual expressions used in visual programming include sketches, icons, diagrams or demonstrations of actions made with graphical objects, etc.

Visual programming emerged from the combination of the following fields: computer graphics, programming languages, and human-machine interaction. There is a misconception that visual programming and visual programming languages aim to eliminate text because the latter include text to some extent in a multidimensional context.

The goals sought to be achieved by programming in LPVs were primarily to improve the speed with which programming tasks are performed, to make programming more accessible to all people, and to improve the accuracy with which people perform programming tasks.

Strategies used in LPVs

Concrete: involves expressing some aspects of the user using particular instances;

Directly: the programmer can use/manipulate an object or a specific value directly, through semantics;

Explicit: some aspects of the semantics are explicit in the environment if they can be specified directly (visually or textually);

Immediate visual response: implies the immediate display of the effects produced by editing the program.

VPL Classifications

- Pure visual languages;
- Programming systems by exemplification;
- Hybrid systems (visual and text);
- Constraint Oriented Systems;
- Form-based systems.

The most important category is pure visual languages, based on visual techniques throughout the programming process.

The programmer works with icons and other graphical representations to create his program which is then debugged and executed also in the visual environment.

The program is never translated into a text-based intermediate language but is compiled directly into its visual representation.

Also, LPV tries to combine visual elements with textual elements.
This is the category of hybrid systems and includes both systems where programs are created visually and then translated into a textual (high-level) language, and systems that use graphics in a textual language

Constraint-oriented systems are known for simulation design. The programmer models the physical objects as objects of the visual environment, imposing constraints designed to copy the behavior of natural laws. This category is also used for the development of graphical user interfaces.

LPVs based on forms are the ones that borrowed the visualization mode but also the programming metaphors from spreadsheets.

COURSE ORGANISATION

1. TEAM PROJECT, 6 points (3 projects x 2 points)

2. PADLET, visual progress, https://padlet.com/, 1 point

3. Research blog, 3 points

ALGORITHM for grading:

- We form teams of 1-3 students
- We find 1-3 research topics per team by VPDA searching on arxiv.com or paperswithcode.com
- We write a blog post on the paper that is most interesting from the last 3 yrs (2020-2023)
- We start implementing a project based on the paper or another agreed topic in VPDA
- We document the Padlet with the team progress

COURSE TECHNOLOGIES

Orange3, for visual programming and data mining

Weights and Biases, for NN optimization

Slicer3D, for medical datasets analysis

Kodular, for rapid mobile phone developments

Padlet, for visual teamwork

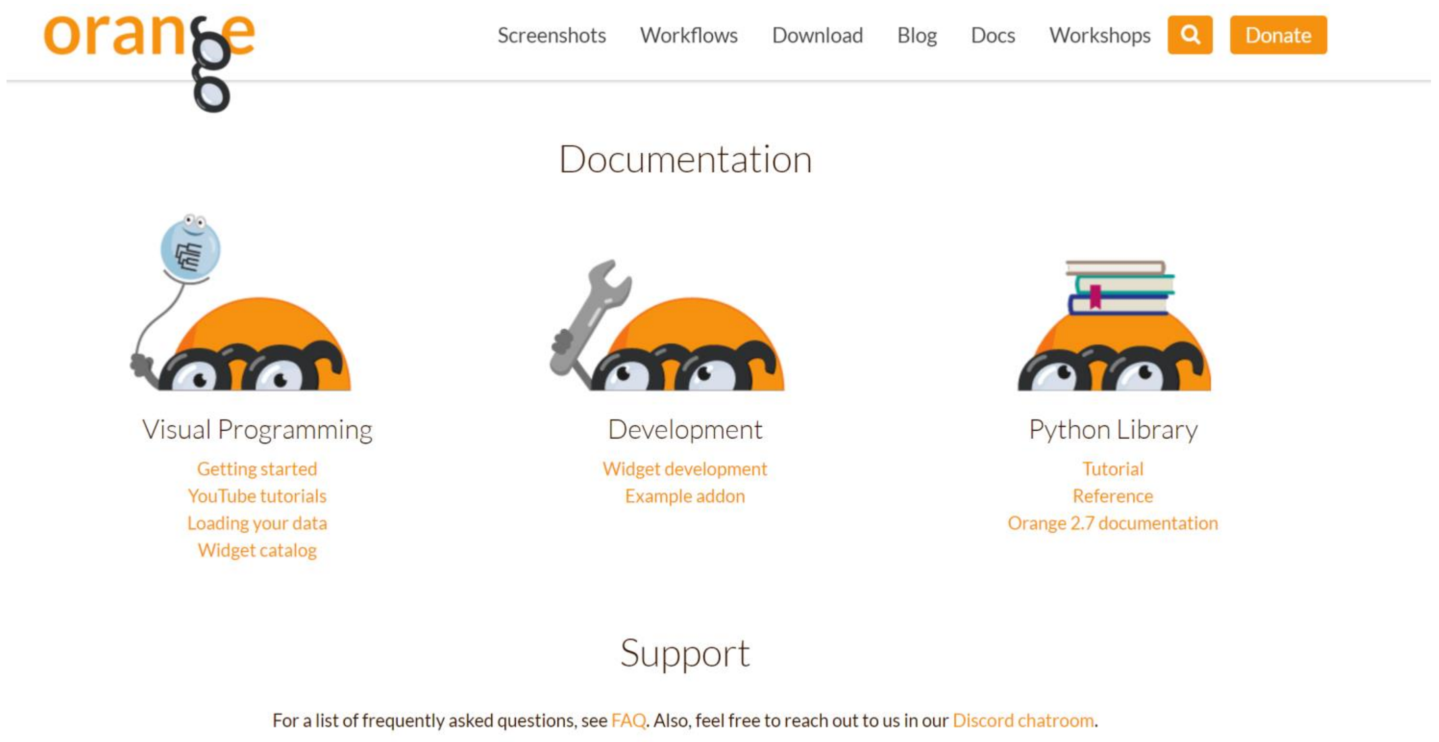
Tableau or alternatives, for visual data analysis

Docker+K8s, for big data processing

D3, for data visualization

Snakemake, for workflow management systems

Orange 3


Screenshots Workflows Download Blog Docs Workshops Donate

Documentation

Visual Programming
Getting started
YouTube tutorials
Loading your data
Widget catalog

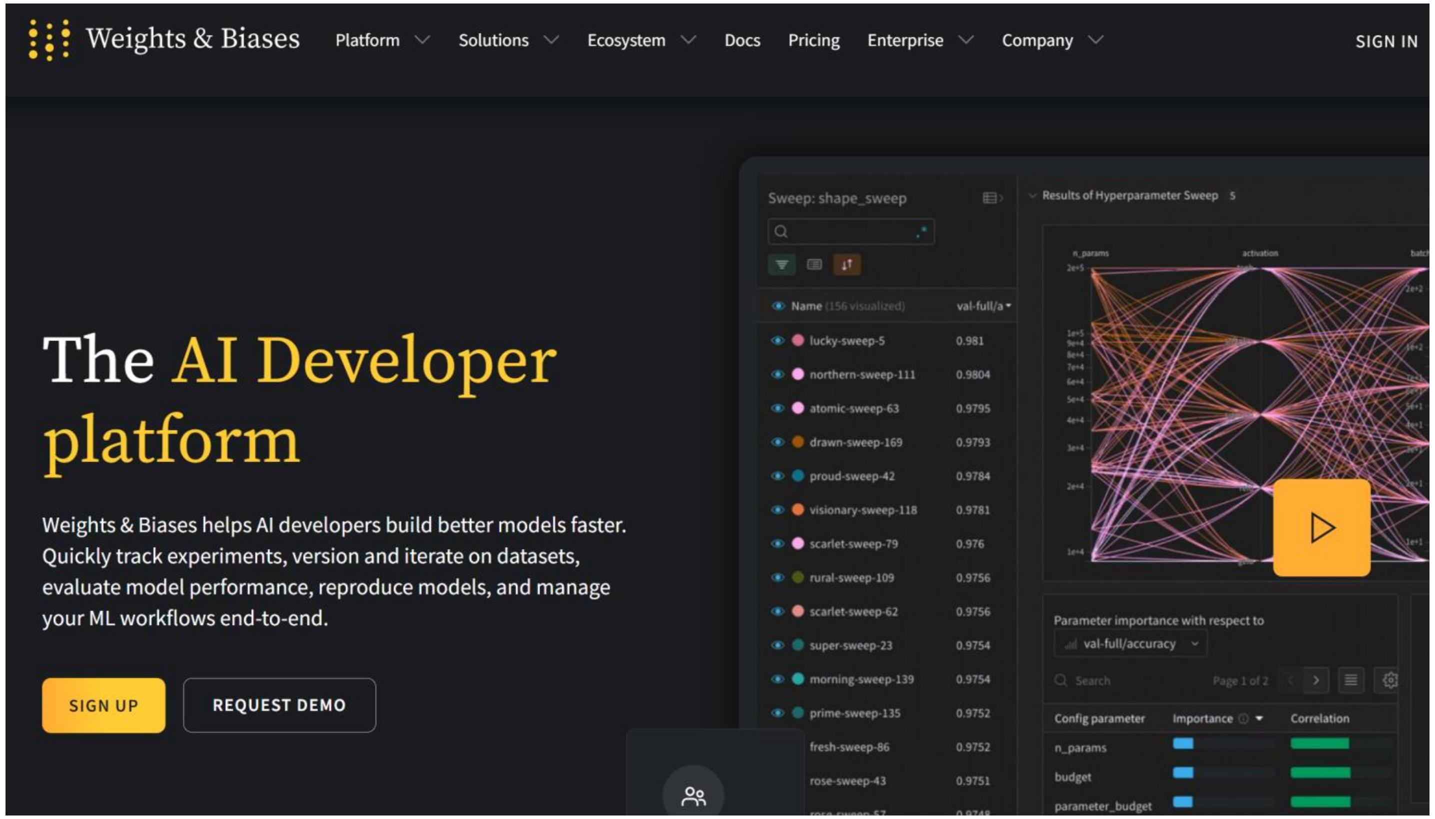
Development
Widget development
Example addon

Python Library
Tutorial
Reference
Orange 2.7 documentation

Support

For a list of frequently asked questions, see FAQ. Also, feel free to reach out to us in our Discord chatroom.

Weights and Biases



The AI Developer platform

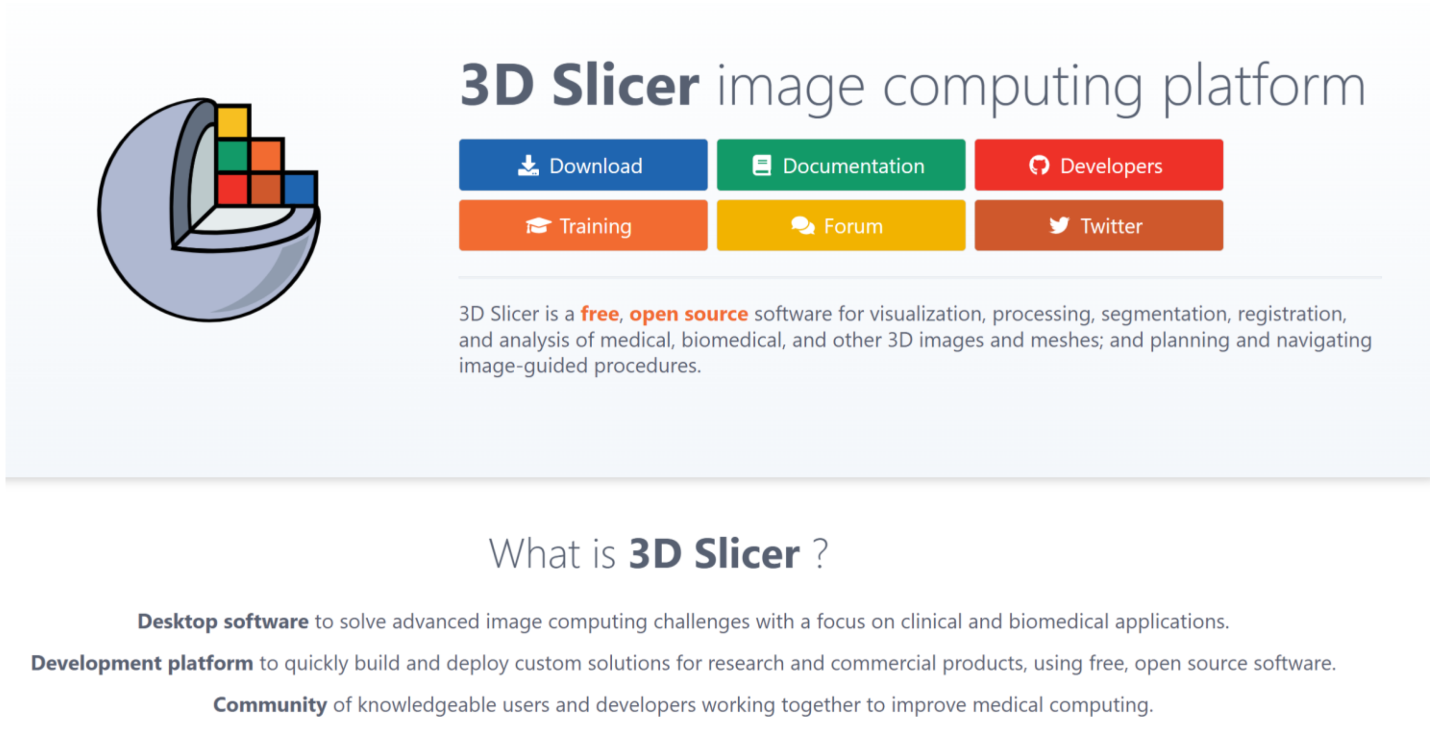
Weights & Biases helps AI developers build better models faster. Quickly track experiments, version and iterate on datasets, evaluate model performance, reproduce models, and manage your ML workflows end-to-end.

Name (156 visualized)	val-full/a
lucky-sweep-5	0.981
northern-sweep-111	0.9804
atomic-sweep-63	0.9795
drawn-sweep-169	0.9793
proud-sweep-42	0.9784
visionary-sweep-118	0.9781
scarlet-sweep-79	0.976
rural-sweep-109	0.9756
scarlet-sweep-62	0.9756
super-sweep-23	0.9754
morning-sweep-139	0.9754
prime-sweep-135	0.9752
fresh-sweep-86	0.9752
rose-sweep-43	0.9751

Parameter importance with respect to val-full/accuracy

Config parameter	Importance	Correlation
n_params		
budget		
parameter_budget		

3D SLICER



3D Slicer image computing platform

3D Slicer is a **free**, **open source** software for visualization, processing, segmentation, registration, and analysis of medical, biomedical, and other 3D images and meshes; and planning and navigating image-guided procedures.

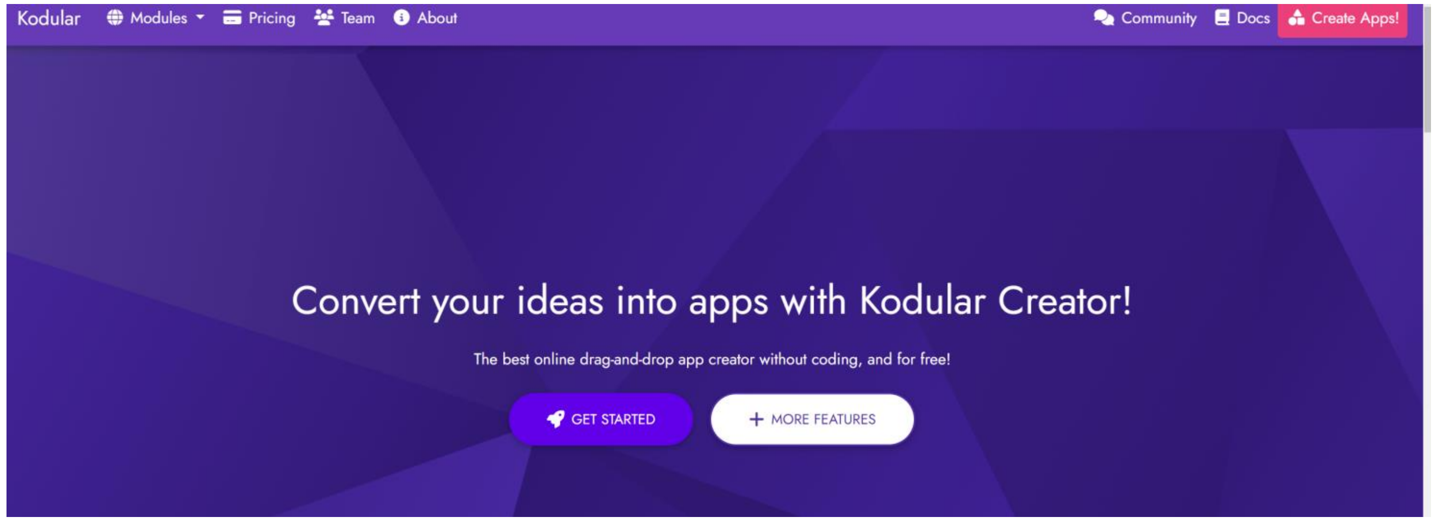
What is **3D Slicer** ?

Desktop software to solve advanced image computing challenges with a focus on clinical and biomedical applications.

Development platform to quickly build and deploy custom solutions for research and commercial products, using free, open source software.

Community of knowledgeable users and developers working together to improve medical computing.

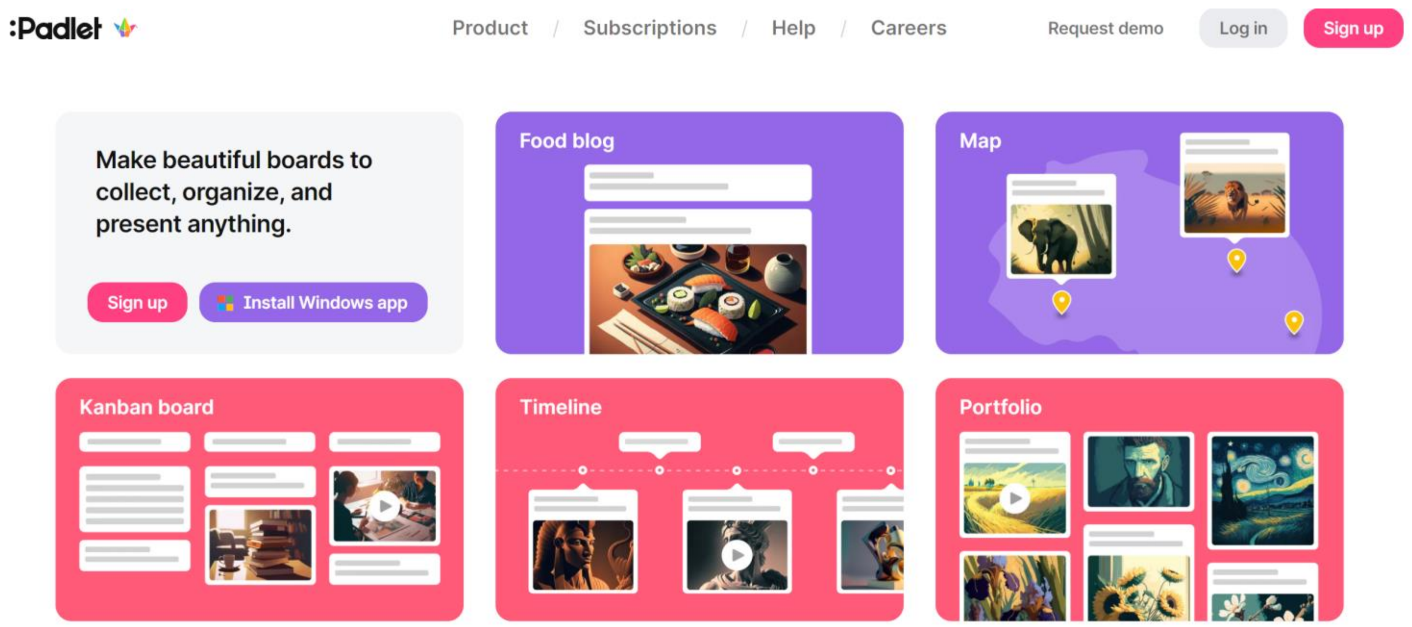
Kodular



Convert your ideas into apps with Kodular Creator!

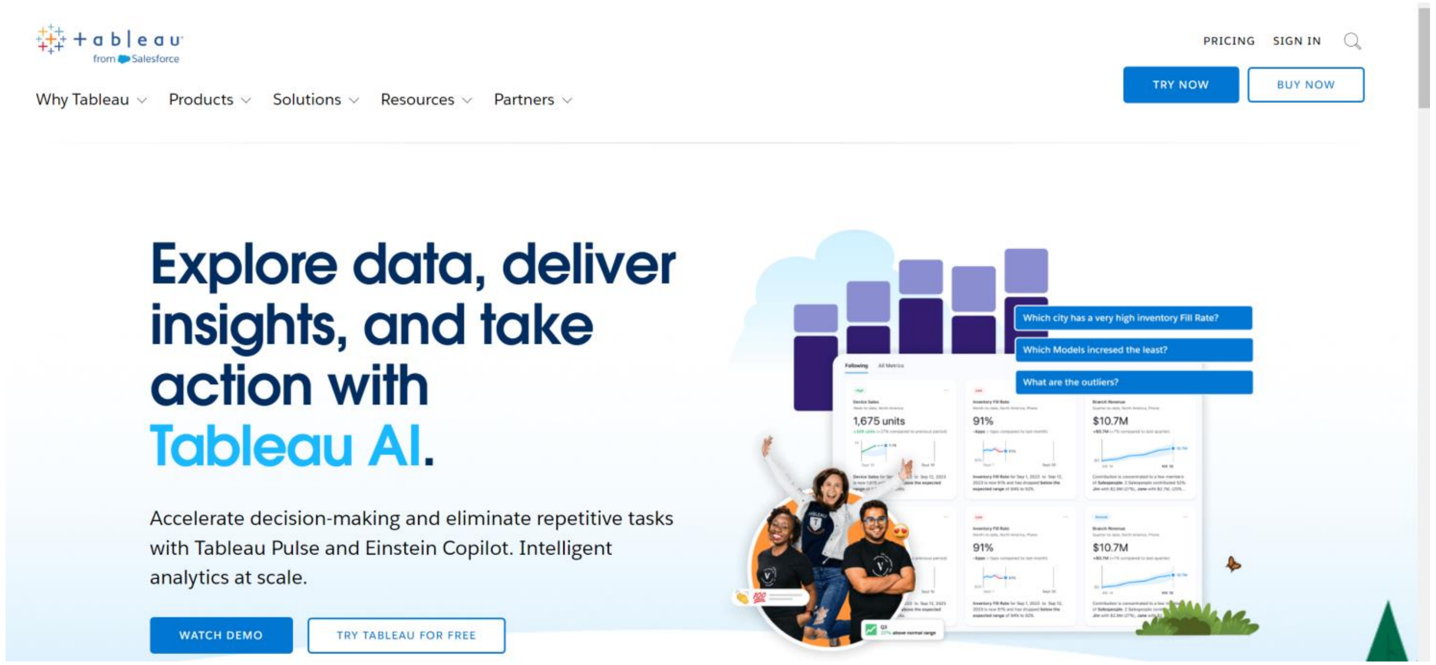
The best online drag-and-drop app creator without coding, and for free!

Padlet



:Padlet
Product / Subscriptions / Help / Careers
Request demo
Log in
Sign up
Make beautiful boards to collect, organize, and present anything.
Sign up
Install Windows app
Food blog
Map
Kanban board
Timeline
Portfolio

Tableau



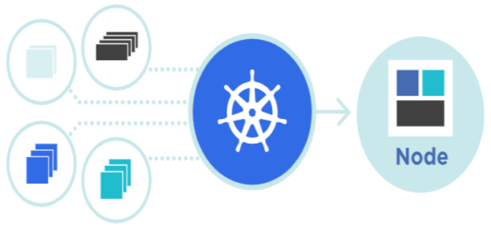
Explore data, deliver insights, and take action with Tableau AI.

Accelerate decision-making and eliminate repetitive tasks with Tableau Pulse and Einstein Copilot. Intelligent analytics at scale.

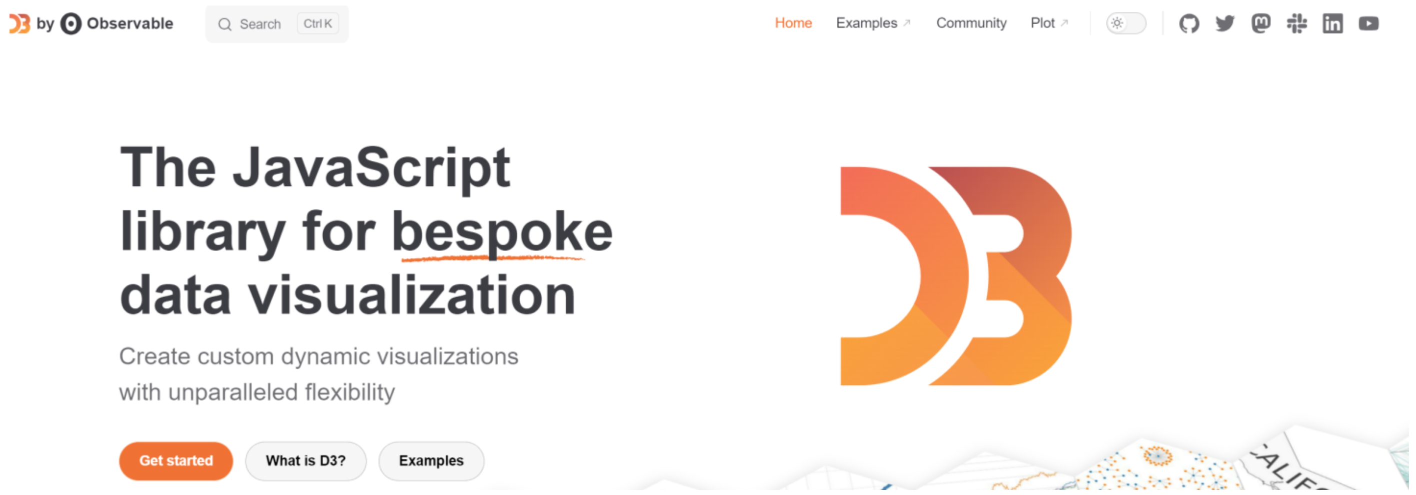
K8S

Kubernetes, also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications.

It groups containers that make up an application into logical units for easy management and discovery. Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community.



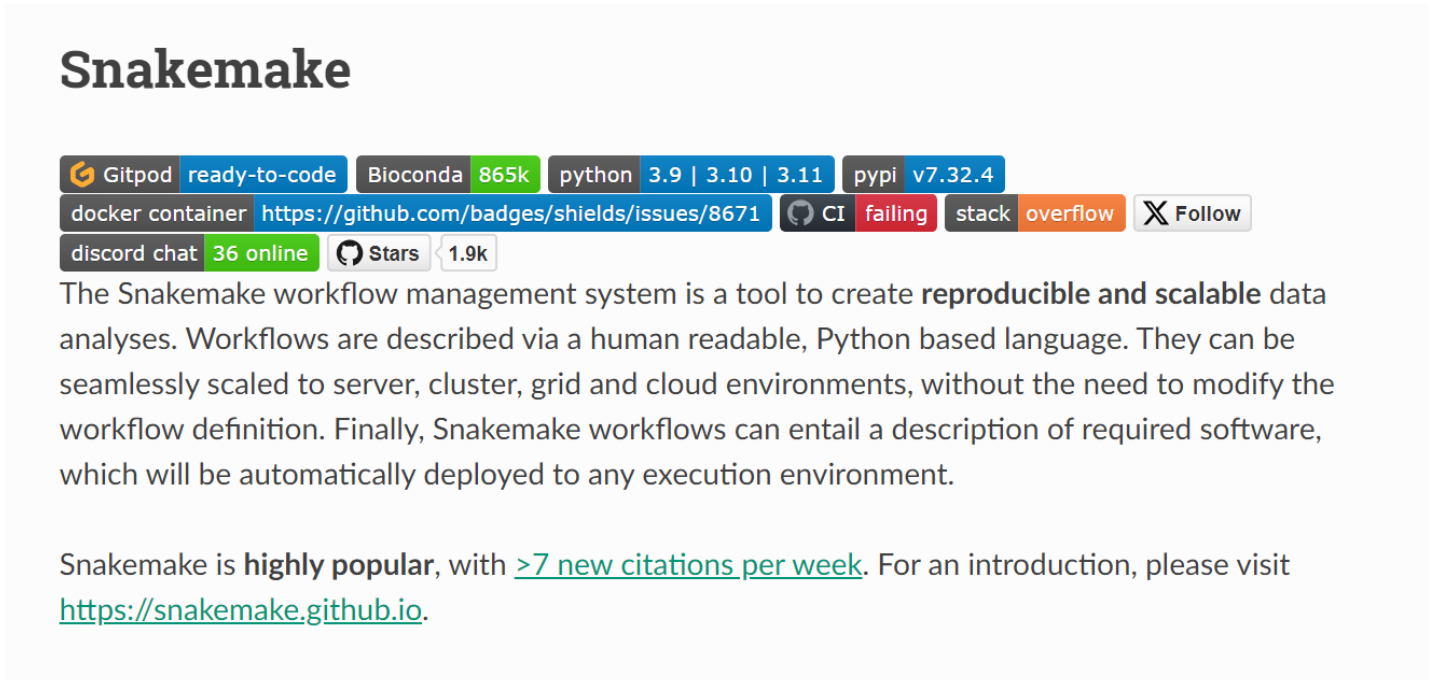
D3



The JavaScript library for bespoke data visualization

Create custom dynamic visualizations with unparalleled flexibility

SNAKEMAKE


Snakemake

Gitpod ready-to-code | Bioconda 865k | python 3.9 | 3.10 | 3.11 | pypi v7.32.4 | docker container https://github.com/badges/shields/issues/8671 | CI failing | stack overflow | Follow | discord chat 36 online | Stars 1.9k

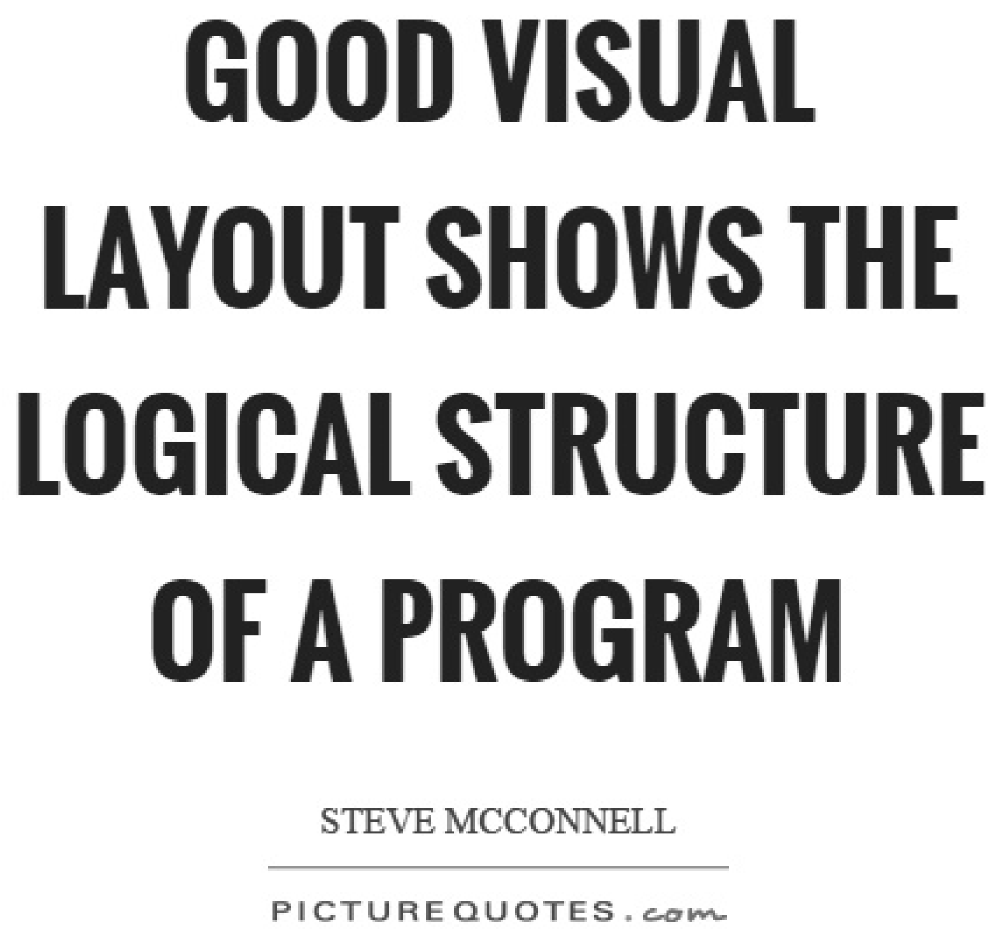
The Snakemake workflow management system is a tool to create **reproducible and scalable** data analyses. Workflows are described via a human readable, Python based language. They can be seamlessly scaled to server, cluster, grid and cloud environments, without the need to modify the workflow definition. Finally, Snakemake workflows can entail a description of required software, which will be automatically deployed to any execution environment.

Snakemake is **highly popular**, with >7 new citations per week. For an introduction, please visit https://snakemake.github.io.

- BLOG POSTS: https://vdpa23.blogspot.com/

- Access via email of team representative

- Do not forget UML



GOOD VISUAL LAYOUT SHOWS THE LOGICAL STRUCTURE OF A PROGRAM

STEVE MCCONNELL

PICTUREQUOTES.com