



GDC-CADD

# An RDKit-based JS component for molecule visualization

Paolo Tosco, Nico Pulver, Nik Stiefl  
10<sup>th</sup> RDKit UGM  
October 15, 2021

# Agenda

Motivation

Implementation

Live demo

Outlook



# Motivation

# Visualize 2D molecules in the browser: server-based

- At NIBR we have several web apps that require 2D molecule visualization
- Currently this is handled by a React component
- The web client requests a 2D layout to the server (SMILES or MDL molblock)
- The server generates the 2D layout and sends it back to the web client as a PNG image
- A number of configuration options is available
  - Width, height
  - Align the layout to a scaffold (SMILES or MDL molblock)
  - Font, coloring, etc.

# Visualize 2D molecules in the browser: client-based

- We would like to switch from a server-based app to a client-based app for performance reasons
- We would like to move from a static PNG image to a vector image that can be easily rescaled without losing definition (HTML5 canvas, SVG)
- We would like the app to be written in vanilla JavaScript to be independent from a specific technology (i.e., not be bound to React)
- Wrappers for different technologies can be easily built around this low-level app

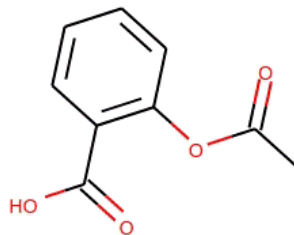
# RDKit in the browser

- 2016: at the UGM Guillaume Godin presented the first JavaScript port of RDKit through emscripten
- 2019: Greg Landrum introduced the new RDKit JavaScript wrappers

<http://rdkit.blogspot.com/2019/11/introducing-new-rdkit-javascript.html>

A subset of RDKit was ported to WebAssembly through emscripten

## RDKit-JS demo



CC(=O)Oc1ccccc1C(=O)O

SMILES:  SMARTS:

## Computed values

AMW: 180.159

MolLogP: 1.3101

MFP2:

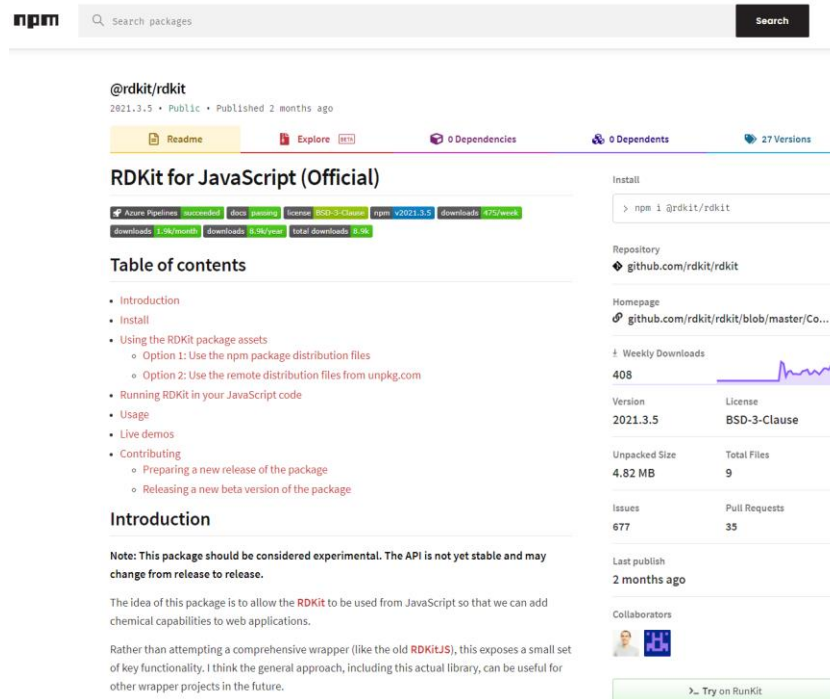
001001000011010000000001000000000100000100000001001000010000100011000001100

# RDKit in the browser

- 2021: Michel Moreau made the RDKit JavaScript MinimalLib available as a npm package

<https://www.npmjs.com/package/@rdkit/rdkit>

This further lowers the barrier to access RDKit functionality in the browser



The screenshot shows the npm package page for `@rdkit/rdkit`. The package is version 2021.3.5, published 2 months ago, and is public. It has 0 dependencies, 0 dependents, and 27 versions. The package is titled "RDKit for JavaScript (Official)". The table of contents includes: Introduction, Install, Using the RDKit package assets (Option 1: Use the npm package distribution files, Option 2: Use the remote distribution files from unpkg.com), Running RDKit in your JavaScript code, Usage, Live demos, and Contributing (Preparing a new release of the package, Releasing a new beta version of the package). The introduction notes that the package is experimental and the API is not yet stable. The package is available on GitHub at `github.com/rdkit/rdkit`. The weekly downloads are 408. The license is BSD-3-Clause. The unpacked size is 4.82 MB and there are 9 total files. There are 677 issues and 35 pull requests. The last publish was 2 months ago. The collaborators are Michel Moreau and another person. A button at the bottom says "Try on RunKit".

npm Search packages Search

@rdkit/rdkit  
2021.3.5 • Public • Published 2 months ago

Readme Explore 0 Dependencies 0 Dependents 27 Versions

RDKit for JavaScript (Official)

Azure Pipelines Successful Docs Passing License BSD-3-Clause npm v2021.3.5 downloads 475/week  
downloads 1.9k/week downloads 9.9k/year total downloads 6.9k

Table of contents

- Introduction
- Install
- Using the RDKit package assets
  - Option 1: Use the npm package distribution files
  - Option 2: Use the remote distribution files from unpkg.com
- Running RDKit in your JavaScript code
- Usage
- Live demos
- Contributing
  - Preparing a new release of the package
  - Releasing a new beta version of the package

Introduction

Note: This package should be considered experimental. The API is not yet stable and may change from release to release.

The idea of this package is to allow the **RDKit** to be used from JavaScript so that we can add chemical capabilities to web applications.

Rather than attempting a comprehensive wrapper (like the old **RDKitJS**), this exposes a small set of key functionality. I think the general approach, including this actual library, can be useful for other wrapper projects in the future.

Install

```
> npm i @rdkit/rdkit
```

Repository  
github.com/rdkit/rdkit

Homepage  
github.com/rdkit/rdkit/blob/master/Co...

Weekly Downloads  
408

Version 2021.3.5 License BSD-3-Clause

Unpacked Size 4.82 MB Total Files 9

Issues 677 Pull Requests 35

Last publish 2 months ago

Collaborators

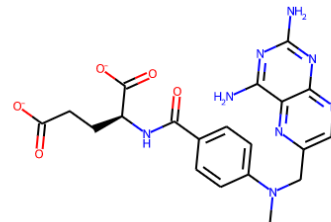
Try on RunKit

# Web apps using RDKit MinimalLib

- Building web apps dealing with chemistry becomes easier

<http://chembl.blogspot.com/2021/03/target-predictions-in-browser-with.html>

Multitask Target prediction with RDKit MinimalLib (JS) and ONNX.js



SMILES

```
CN(Cc1cnc2nc(N)nc(N)c2n1)c1ccc(C(=O)N[C@@H](CCC(=O)[O-])C(=O)[O-])cc1
```

Predictions

Search



Target ChEMBL id	proba ↓
▼ CHEMBL4409	0.9893653988838196
▼ CHEMBL4191	0.9790000319480896
▼ CHEMBL1744525	0.9765774607658386



# Web apps using RDKit MinimalLib

- Datagrok
- Web-based data analysis tool
- Uses RDKit MinimalLib for 2D molecule visualization

<https://datagrok.ai>

The screenshot displays the Datagrok web application interface. The browser address bar shows the URL [https://datagrok.ai/ChEMBL\\_quinolones\\_DG](https://datagrok.ai/ChEMBL_quinolones_DG). The application features a sidebar on the left with various tool icons for search, viewers, columns, layouts, actions, models, and algorithms. The main workspace contains a table with the following columns: col 1, Smiles, Scaffold, ChEMBL ID, Name, and Synonyms. The table lists several quinolone compounds, with the first row highlighted in green.

col 1	Smiles	Scaffold	ChEMBL ID	Name	Synonyms
17			CHEMBL3409681		
18			CHEMBL4204054		
19			CHEMBL2106745	NORFLOXACIN SUCCINIL	NORFLOXACIN
20			CHEMBL2106560	FANDOFLOXACIN	FANDOFLOXACIN
21			CHEMBL2104028	AMIFLOXACIN MESYLATE	AMIFLOXACIN M
22			CHEMBL4202566		

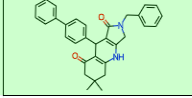
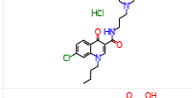
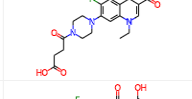
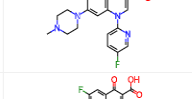
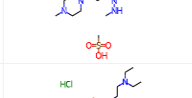
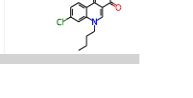
# Web apps using RDKit MinimalLib

- Molecules can be aligned to a scaffold if the scaffold matches

Screenshot of a web application interface for ChEMBL quinolones data, showing a table of molecules and their scaffolds.

URL: [https://datagrok.ai/ChEMBL\\_quinolones\\_DG](https://datagrok.ai/ChEMBL_quinolones_DG)

Table Columns: col 1, Smiles, Scaffold, ChEMBL ID, Name, Synonyms

col 1	Smiles	Scaffold	ChEMBL ID	Name	Synonyms
17	16		CHEMBL3409681		
18	17		CHEMBL4204054		
19	18		CHEMBL2106745	NORFLOXACIN SUCCINIL	NORFLOXACIN
20	19		CHEMBL2106560	FANDOFLOXACIN	FANDOFLOXACIN
21	20		CHEMBL2104028	AMIFLOXACIN MESYLATE	AMIFLOXACIN M
22	21		CHEMBL4202566		

# Web apps using RDKit MinimalLib

- Molecules can be aligned to a scaffold if the scaffold matches
- The scaffold atoms can be highlighted

Screenshot of a web application interface showing a table of chemical data. The URL is [https://datagrok.ai/ChEMBL\\_quinolones\\_DG](https://datagrok.ai/ChEMBL_quinolones_DG). The table displays columns: col 1, Smiles, Scaffold, ChEMBL ID, Name, and Synonyms. The first row (17) is highlighted in green, showing a complex molecule structure and its corresponding scaffold (a quinolone core). The subsequent rows (18-22) show various quinolone derivatives, including Norfloxacin, Fandofloxacin, and Amifloxacin, with their respective structures and scaffolds.







col 1	Smiles	Scaffold	ChEMBL ID	Name	Synonyms
17			CHEMBL3409681		
18			CHEMBL4204054		
19			CHEMBL2106745	NORFLOXACIN SUCCINIL	NORFLOXACIN
20			CHEMBL2106560	FANDOFLOXACIN	FANDOFLOXACIN
21			CHEMBL2104028	AMIFLOXACIN MESYLATE	AMIFLOXACIN M
22			CHEMBL4202566		

# Web apps using RDKit MinimalLib

- Have better granularity
- Align or highlight the scaffold only for selected molecules
- Communicate the per-molecule setting to the parent app for storage

Screenshot of a web application interface for managing chemical data, specifically a table of quinolones. The URL is [https://datagrok.ai/ChEMBL\\_quinolones\\_DG](https://datagrok.ai/ChEMBL_quinolones_DG).

The interface includes a sidebar with navigation options (Search, Viewers, Columns, Layouts, Actions, Models, Algorithms) and a main table displaying chemical data. The table has columns for Smiles, Scaffold, ChEMBL ID, Name, and Synonyms. The first row is highlighted in green.

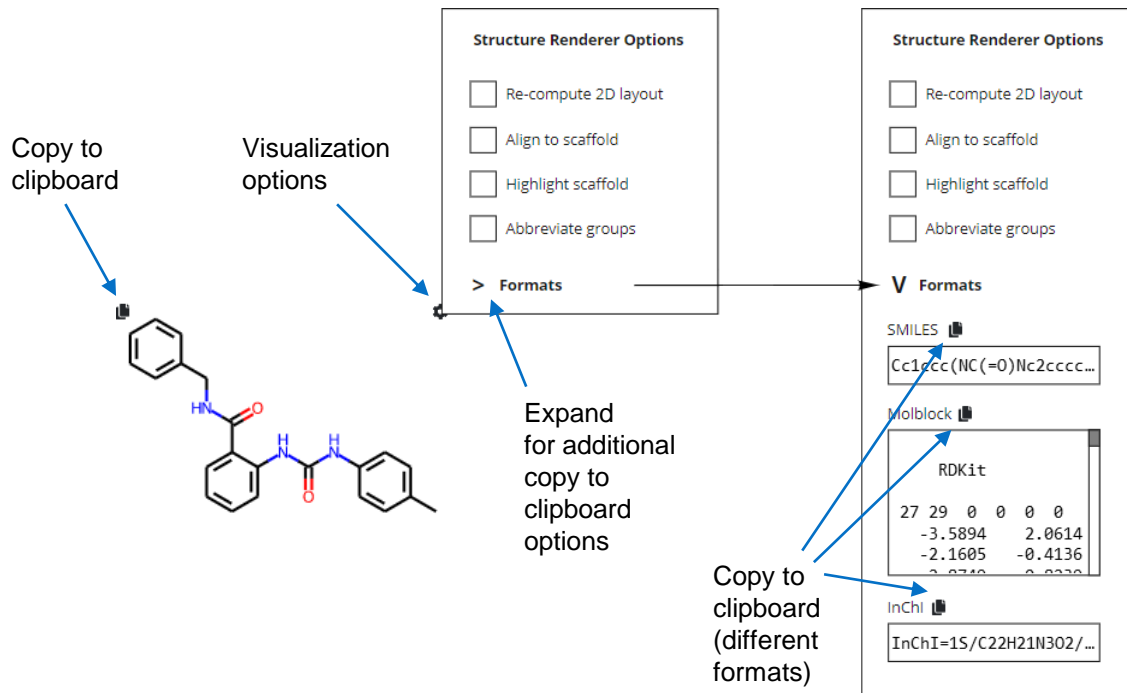
col 1	Smiles	Scaffold	ChEMBL ID	Name	Synonyms
17	16		CHEMBL3409681		
18	17		CHEMBL4204054		
19	18		CHEMBL2106745	NORFLOXACIN SUCCINIL	NORFLOXACIN
20	19		CHEMBL2106560	FANDOFLOXACIN	FANDOFLOXACIN
21	20		CHEMBL2104028	AMIFLOXACIN MESYLATE	AMIFLOXACIN M
22	21		CHEMBL4202566		



# Implementation

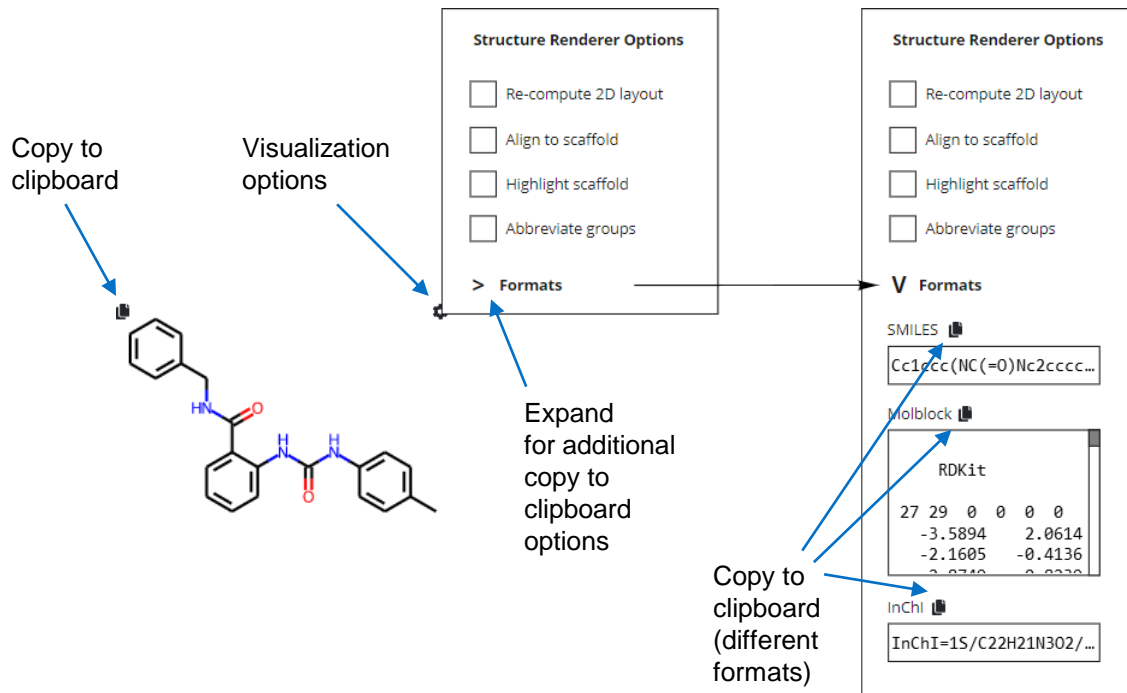
# StructureRenderer.js design

- Each 2D layout is decorated with two icons that appear upon hovering
- Multiple checkboxes with visualization options
- Multiple formats can be copied to clipboard



# StructureRenderer.js design

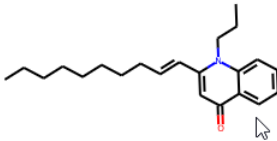
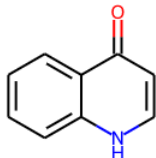
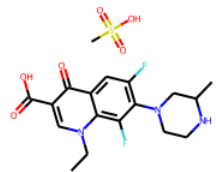
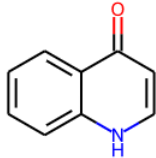
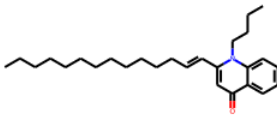
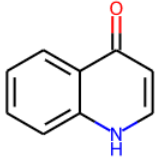
- Having no JS dependencies other than the RDKit MinimalLib, this should work in something as simple as, e.g., a Jupyter Notebook



# Pandas DataFrame in Jupyter

- Most of us are familiar with the visualization of a pandas DataFrame containing RDKit molecules in Jupyter

```
PandasTools.ChangeMoleculeRendering(df_quinolones)  
show(df_quinolones)
```

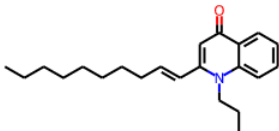
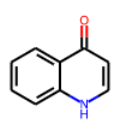
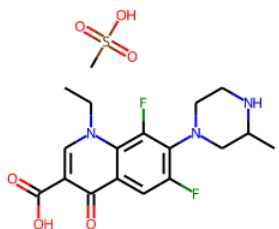
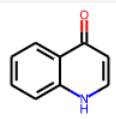
Smiles	Scaffold	ChEMBL ID	Name	Synonyms
		CHEMBL1782626	NaN	NaN
		CHEMBL2106827	LOMEFLOXACIN MESYLATE	LOMEFLOXACIN MESYLATE[SC-471118
		CHEMBL1782648	NaN	NaN



# Pandas DataFrame in Jupyter with StructRenderer.js

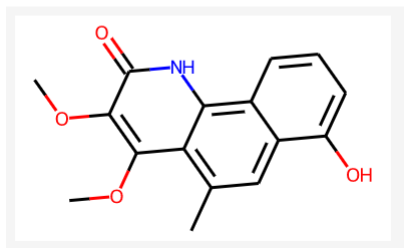
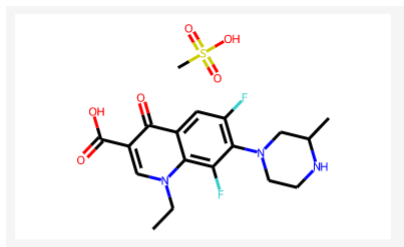
```
PandasTools.ChangeMoleculeRendering(df_quinolones)  
show(df_quinolones)
```

- Where's the difference?
- They look pretty much the same

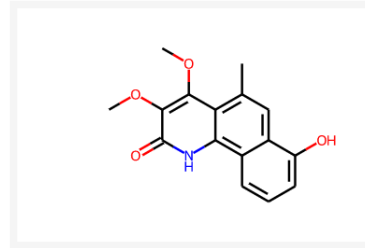
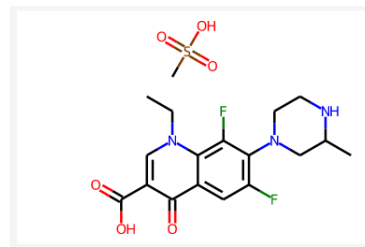
Smiles	Scaffold	ChEMBL ID	Name	Synonyms
		CHEMBL1782626	NaN	NaN
		CHEMBL2106827	LOMEFLOXACIN MESYLATE	LOMEFLOXACIN MESYLATE[SC-471118]

# Molecules are rotated to look prettier

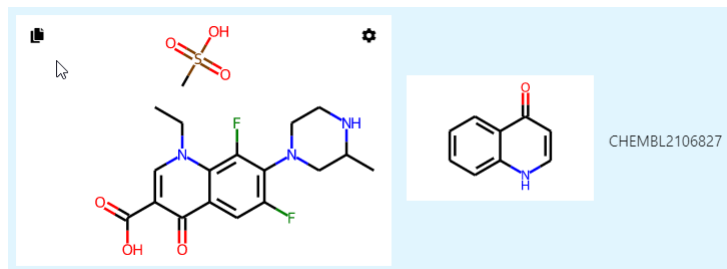
RDKit aligns principal axes to Cartesian axes to exploit available space at best



StructureRenderer.js canonicalizes 2D coordinates and then rotates molecules such that most bonds have a 30-degree angle with the X axis



# Additional controls appear on hover



Copy to  
clipboard from  
Jupyter and  
paste into  
different tools

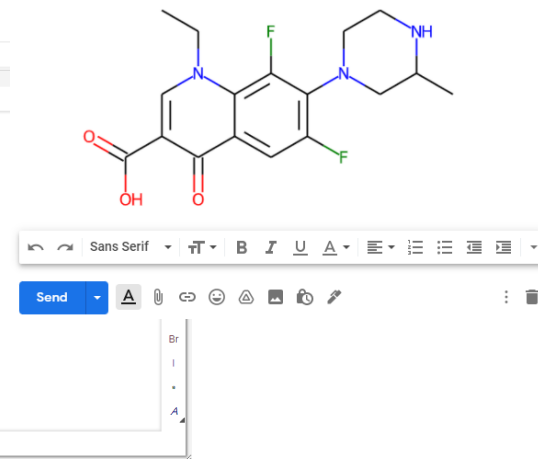
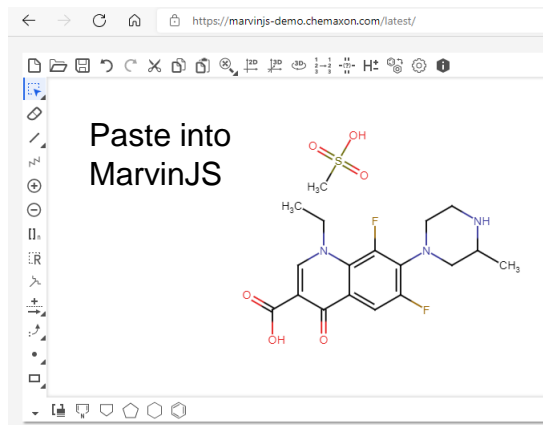
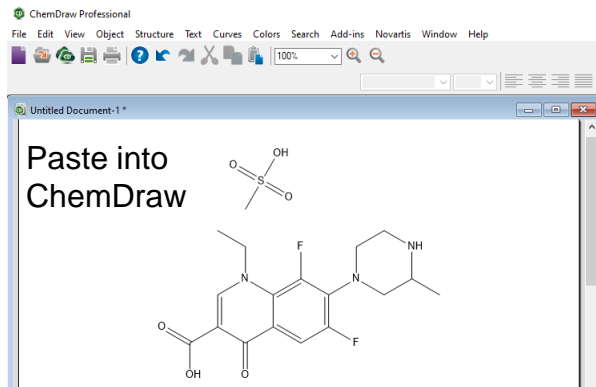
How cool is RDKit 2D depiction?

Landrum Gregory

How cool is RDKit 2D depiction?

Hi Greg,

Paste into  
e-mail (or  
PowerPoint)



# Tailor the appearance of the 2D layout

- The appearance of the 2D layout can be tailored through multiple, non-exclusive options

The screenshot displays the JupyterLab environment with the `StructRenderer.ipynb` notebook open. The interface includes a file browser on the left, a top menu bar, and a main workspace. A **Structure Renderer Options** dialog box is open, showing the following options:

- ☐ Re-compute 2D layout
- ☐ Align to scaffold
- ☐ Highlight scaffold
- ☐ Abbreviate groups
- ☐ Add atom indices
- > Formats**

The workspace shows two chemical structures:

- A large, complex molecule (CHEMBL1782626) with a long alkyl chain and a sulfonamide group.
- A smaller, simpler molecule (CHEMBL2106827) with a quinoline core.

# Re-compute 2D layout

- The layout can be re-computed using CoordGen (useful in case of, e.g., macrocycles or poor initial coordinates if the molecule was generated from a MDL molblock)

The screenshot displays the JupyterLab web interface at <https://localhost:9900/lab>. The left sidebar shows a file explorer with a 'snippets' folder containing 'StructRenderer.ipynb' and 'Untitled.ipynb'. The main workspace shows a chemical structure editor with a long-chain molecule. A context menu titled 'Structure Renderer Options' is open, showing the following options:

- ☒ Re-compute 2D layout
- ☐ Align to scaffold
- ☐ Highlight scaffold
- ☐ Abbreviate groups
- ☐ Add atom indices
- > Formats

Below the menu, two chemical structures are displayed in a grid. The top structure is a complex molecule with a sulfonamide group, labeled 'CHEMBL1782626'. The bottom structure is a quinoline derivative, labeled 'CHEMBL2106827'.

# Align to scaffold

- The layout can be aligned to a scaffold (as previously seen in the Datagrok example)

The screenshot displays the JupyterLab web interface at <https://localhost:9900/lab>. The left sidebar shows a file explorer with a 'snippets' folder containing 'StructRenderer.ipynb' and 'Untitled.ipynb'. The main workspace shows a chemical structure editor with a long-chain molecule. A context menu titled 'Structure Renderer Options' is open, showing the following options:

- ☐ Re-compute 2D layout
- ☒ Align to scaffold
- ☐ Highlight scaffold
- ☐ Abbreviate groups
- ☐ Add atom indices
- [Formats](#)

Below the menu, two chemical structures are displayed in separate panels. The left panel shows a complex molecule with a quinoline core, fluorine atoms, and a sulfonamide group. The right panel shows a simpler quinoline derivative. The rightmost column of the interface lists chemical identifiers: 'CHEMBL1782626' and 'CHEMBL2106827'.

# Align to scaffold with highlighting

- We can also highlight the scaffold atoms in the molecule

The screenshot displays a JupyterLab environment with a web browser address bar showing `https://localhost:9900/lab`. The interface includes a file explorer on the left with a sidebar containing icons for file operations. The main workspace shows a Jupyter notebook titled `StructRenderer.ipynb` with a code cell containing a chemical structure. A context menu titled **Structure Renderer Options** is open over the structure, listing the following options:

- ☐ Re-compute 2D layout
- ☒ Align to scaffold
- ☒ Highlight scaffold
- ☐ Abbreviate groups
- ☐ Add atom indices
- > Formats**

The chemical structure in the code cell is a complex molecule with a highlighted scaffold. Below the code cell, a preview window shows the rendered structure, which is a complex molecule with a highlighted scaffold. To the right of the preview window, the identifier `CHEMBL1782626` is displayed. Below the preview window, another chemical structure is shown, which is a quinoline derivative, with the identifier `CHEMBL2106827` displayed to its right.

# Display abbreviated functional groups

- Functional groups can be displayed in their abbreviated form

The screenshot displays a JupyterLab environment with a web browser interface. The address bar shows 'https://localhost:9900/lab'. The left sidebar contains a file explorer with a 'snippets' folder and files 'StructRenderer.ipynb' and 'Untitled.ipynb'. The main area shows a chemical structure editor with a complex molecule. A 'Structure Renderer Options' dialog box is open, listing the following settings:

- ☐ Re-compute 2D layout
- ☒ Align to scaffold
- ☒ Highlight scaffold
- ☒ Abbreviate groups
- ☐ Add atom indices
- > Formats

Below the dialog, two chemical structures are shown. The left structure is a complex molecule with a quinoline core, fluorine atoms, a carboxylic acid group, and a sulfonic acid group. The right structure is a simpler molecule, a quinoline derivative, with the identifier 'CHEMBL2106827' next to it. The top structure is identified by 'CHEMBL1782626'.



# Unchecking options works

- Unchecking individual options triggers a layout update to only apply the relevant settings

The screenshot shows the JupyterLab interface with the 'Structure Renderer Options' dialog box open. The dialog box contains the following options:

- ☐ Re-compute 2D layout
- ☐ Align to scaffold
- ☒ Highlight scaffold
- ☒ Abbreviate groups
- ☐ Add atom indices
- > Formats**

The background shows a chemical structure editor with a complex molecule and two smaller molecules (CHEMBL1782626 and CHEMBL2106827) displayed below it.

# Bonus option: add atom indices

- Compared to the original design I added the "Add atom indices" option that I find very handy when working with molecules in Jupyter

The screenshot shows a JupyterLab environment with a file browser on the left containing 'snippets', 'StructRenderer.ipynb', and 'Untitled.ipynb'. The main area displays a chemical structure editor with a long-chain molecule at the top and two more complex molecules below. A context menu titled 'Structure Renderer Options' is open, showing the following options:

- ☐ Re-compute 2D layout
- ☐ Align to scaffold
- ☐ Highlight scaffold
- ☐ Abbreviate groups
- ☒ Add atom indices
- > Formats

The molecules shown are:

- Top: A long-chain molecule with a terminal amine group, identified as CHEMBL1782626.
- Bottom Left: A complex molecule with multiple functional groups and atom indices (1-20), identified as CHEMBL2106827.
- Bottom Right: A quinoline derivative with atom indices (1-13).

# Copy follows WYSYWYG paradigm

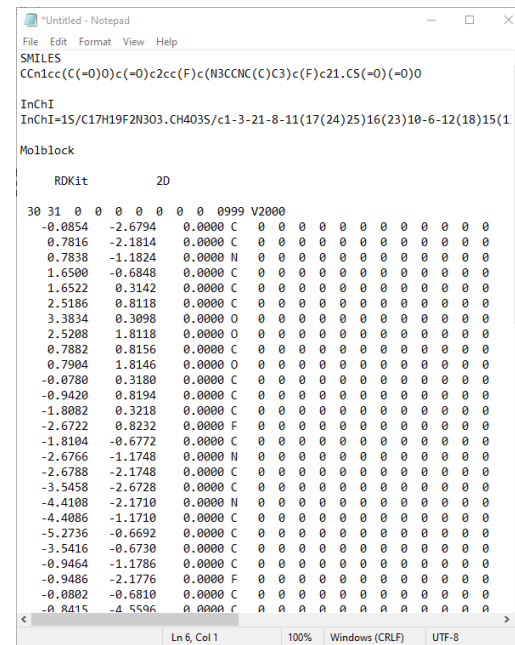
- The copy functionality puts on the clipboard the same image/coordinates as shown in Jupyter

The screenshot displays a Jupyter Notebook environment with a file named 'StructRenderer.ipynb'. The 'Code' tab is active, showing a chemical structure of a long-chain alkylamine. A 'Structure Renderer Options' menu is open, listing the following settings:

- ☐ Re-compute 2D layout
- ☒ Align to scaffold
- ☒ Highlight scaffold
- ☐ Abbreviate groups
- ☒ Add atom indices
- [Formats](#)

Below the menu, a chemical structure of a complex molecule is shown with atom indices (1-20) and a highlighted scaffold. To the right, a web browser window displays the URL <https://marvinjs-demo.chemaxon.com/latest/>. The browser shows a chemical structure editor with a toolbar and a canvas displaying a chemical structure. On the far right, a chat window titled 'Doesn't it look even better with atom indices?' shows a conversation with 'Landrum Gregory' and 'Hi Greg.' followed by a chemical structure with atom indices.

- Expanding "Formats" gives access to further paste formats (SMILES, InChI, MDL molblock)





**Live demo**



# Outlook

# Outlook

- We have realized a prototype vanilla JS app to support RDKit-based 2D molecule visualization in the browser
- The only dependency is on RDKit MinimalLib
- This makes the app lightweight and allows adoption in many different contexts
- We are planning to release the app as open-source code on GitHub

# Acknowledgments

- NIBR
  - Grégori Gerebtzoff
  - Andreas Gasser
  - Nik Clare
  - Robert Strulak
  - Riccardo Vianello
  - Jimmy Kromann
  - [...]
- EPAM
  - Dmitrii Makarkin
  - Ekaterina Stepanova
  - Anastasia Lebedeva
  - [...]
- RDKit
  - Greg Landrum
  - David Cosgrove
  - Brian Kelley
  - [...]
- Datagrok
  - Dan Skatov
  - Alex Paramanov
  - Andrew Skalkin
  - [...]
- Schrödinger
  - Nic Zonta
  - Rachel Walker
  - Ricardo Rodríguez
  - [...]





**Thank you**