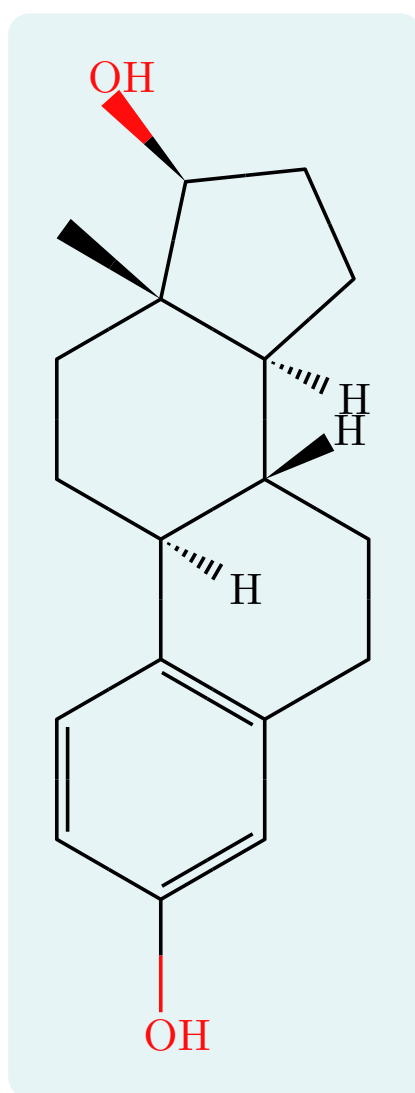


typed-smiles

Render SMILES strings as 2D molecular diagrams

User Guide



Version 0.3.0

Contents


1	Introduction	2
2	Getting started	3
2.1	Installation and import	3
2.2	Your first molecule	3
3	The <code>smiles()</code> function	4
3.1	Argument reference	4
3.2	<code>smiles-str</code>	5
3.3	<code>scale</code>	5
3.4	<code>bond-length</code>	5
3.5	<code>font-size</code>	6
3.6	<code>font</code>	6
3.7	<code>bond-stroke</code>	7
3.8	<code>color</code>	7
3.9	<code>rotation</code>	8
3.10	<code>show-all-h</code>	8
4	Atoms and charges	9
4.1	Aromatic rings	9
4.2	Charges	9
5	Hydrogens	10
5.1	Default display	10
5.2	Label orientation	10
6	Stereochemistry	11
6.1	Tetrahedral centers: <code>@</code> and <code>@@</code>	11
6.2	Double bonds: <code>/</code> and <code>\</code>	12
6.3	Manual wedges: <code>!w</code> and <code>!h</code>	12
7	Colors	13
7.1	Jmol CPK palette	13
7.2	Custom colors (<code>atom-colors</code>)	13
7.3	Label colors (<code>{label style}</code>)	15
8	Abbreviated groups	17
8.1	Plain labels	17
8.2	Colored labels	17
9	Chemical formulas: <code>ce()</code>	18
9.1	Fonts and size	18
10	Reaction schemes	19
10.1	<code>mol()</code>	19
10.2	<code>rxn-arrow()</code>	19
10.3	<code>reaction()</code>	19
11	Project-wide defaults	23
12	Limitations	24
13	Quick reference	25
13.1	Syntax	25
13.2	<code>smiles()</code> options	25
13.3	<code>reaction()</code> options	25
13.4	Label color names	25

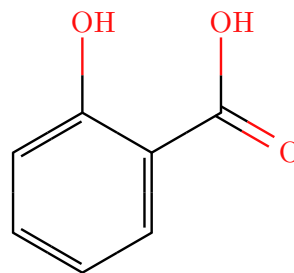
1 Introduction

typed-smiles renders SMILES strings as 2D skeletal molecular diagrams inside Typst documents. A bundled WebAssembly plugin parses the SMILES, computes atom coordinates, implicit hydrogens, and stereochemistry, and the Typst layer draws the bonds, atom labels, colors, charges, hydrogens, wedges, and reaction helpers.

This guide documents every function, argument, and package-specific syntax extension. Each feature comes with a runnable example: the source is on the left, its rendered output on the right.

```
1 #smiles("Oc1ccccc1C(=O)O")
```

 Typst



2 Getting started

2.1 Installation and import

Import the package from the Typst preview namespace. A wildcard import gives you every public symbol:

```
1 #import "@preview/typed-smiles:0.3.0": *
```

 Typst

Or import only what you need:

```
1 #import "@preview/typed-smiles:0.3.0": smiles, ce, mol, rxn-arrow, reaction
```

 Typst


The package exports five symbols:

Symbol	Purpose
<code>smiles</code>	Render a SMILES string as a molecule (the main function).
<code>ce</code>	Chemical formulas and equations, re-exported from <code>chemformula</code> .
<code>mol</code>	Wrap a molecule with an optional caption.
<code>rxn-arrow</code>	A reaction arrow with conditions above and below.
<code>reaction</code>	Lay out a multi-step reaction scheme.

2.2 Your first molecule

Call `smiles` with a SMILES string.

```
1 Ethanol: #smiles("CCO")
```

 Typst


Ethanol:



3 The smiles() function

smiles is the main function. Its full signature is:

```
1  #smiles(  
2    smiles-str,  
3    scale: 1.0,  
4    bond-length: none,  
5    font-size: none,  
6    font: "New Computer Modern",  
7    bond-stroke: none,  
8    color: true,  
9    rotation: 0deg,  
10   show-all-h: false,  
11   atom-colors: (:),  
12  )
```

 Typst

3.1 Argument reference


Argument	Type	Default	Description
smiles-str	str	(required)	The SMILES string.
scale	float	1.0	Balanced scale for bond length, label size, and stroke at once.
bond-length	float / none	none	Bond length factor (1.0 is 30 pt). Overrides <code>scale</code> for length.
font-size	length / none	none	Atom-label font size. Overrides <code>scale</code> for labels.
font	str	"New Computer Modern"	Font family for atom labels.
bond-stroke	length / none	none	Bond stroke width. Overrides <code>scale</code> for strokes.
color	bool	true	Apply Jmol CPK atom colors.
rotation	angle	0deg	Rotate the molecule; labels stay upright.
show-all-h	bool	false	Also label carbon implicit hydrogens.
atom-colors	dictionary	(:)	Color overrides for elements and labels. Element-symbol keys (e.g. <code>0: red</code>) override CPK colors; brace-quoted label keys (e.g. <code>"{PPh3}": purple</code>) override a specific abbreviated group. See Section 7 .

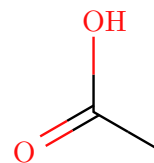
Note. `scale` sizes everything together. Use `bond-length`, `font-size`, and `bond-stroke` to override one dimension on its own; each defaults to `none`, meaning it follows `scale`.

3.2 smiles-str

The positional argument is the SMILES string.

```
1 #smiles("CC(=O)O")
```


 Typst

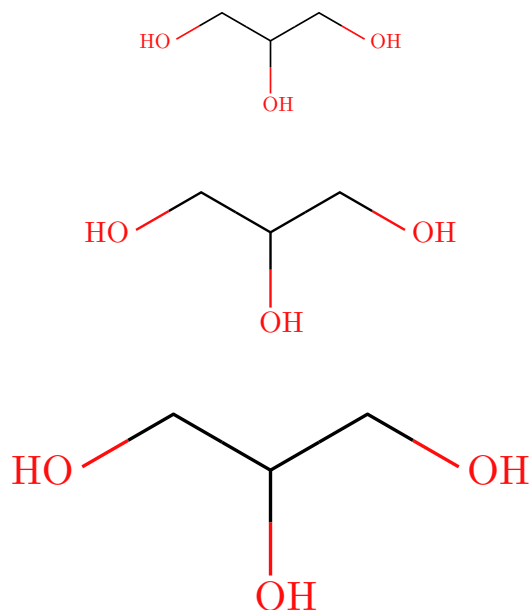


3.3 scale

Scales bond length, labels, and stroke proportionally.

```
1 #smiles("OCC(O)CO", scale: 0.7) \
2 #smiles("OCC(O)CO", scale: 1.0) \
3 #smiles("OCC(O)CO", scale: 1.4)
```


 Typst

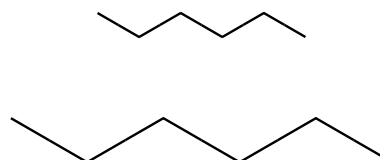


3.4 bond-length

Sets bond length on its own (1.0 is 30 pt per bond), leaving label size and stroke under `scale`.


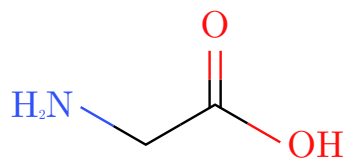
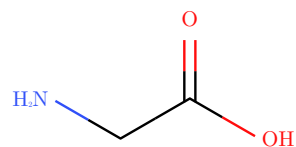
```
1 #smiles("CCCCC", bond-length:
0.6) \
2 #smiles("CCCCC", bond-length: 1.1)
```

 Typst



3.5 font-size


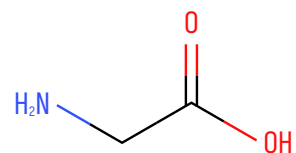
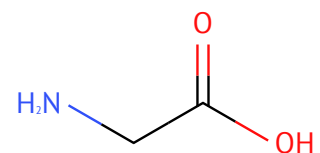
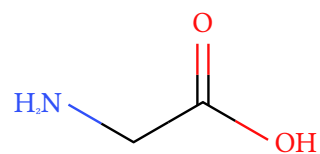
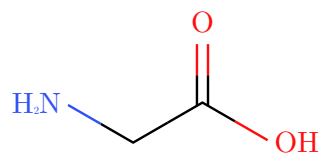
```
1 #smiles("OC(=O)CN", font-size: 8pt) \
2 #smiles("OC(=O)CN", font-size: 14pt)
```

 Typst

3.6 font

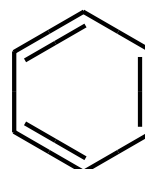
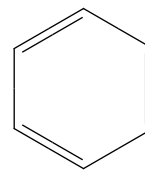
Match the document typeface, or pick something with character.

```
1 #smiles("OC(=O)CN", font: "New
   Computer Modern") \
2 #smiles("OC(=O)CN", font: "Libertinus
   Serif") \
3 #smiles("OC(=O)CN", font: "PT Sans") \
4 #smiles("OC(=O)CN", font: "Iosevka")
```

 Typst

3.7 bond-stroke

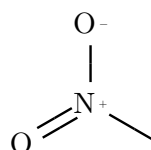
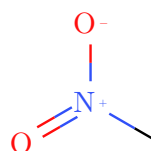
```
1 #smiles("C1=CC=CC=C1", bond-  
  stroke: 0.5pt) \ t Typst  
2 #smiles("C1=CC=CC=C1", bond-stroke: 1.6pt)
```



3.8 color

CPK colors are on by default (see [Section 7](#)). Set `color: false` for a monochrome diagram.


```
1 #smiles("C[N+](=O)[O-]") \ t Typst  
2 #smiles("C[N+](=O)[O-]", color: false)
```

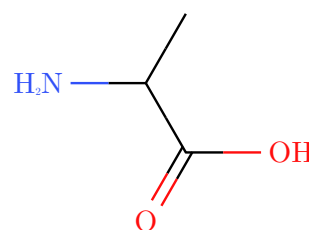
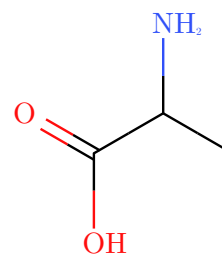


3.9 rotation

Rotates the whole molecule while keeping every label upright.

```
1 #smiles("CC(N)C(=O)O", rotation: 0deg) \
2 #smiles("CC(N)C(=O)O", rotation: 90deg)
```


 Typst

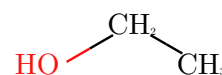


3.10 show-all-h

Hydrogens on heteroatoms are shown by default; carbon hydrogens are hidden. Set `show-all-h: true` to label every implicit hydrogen.

```
1 #smiles("CCO") \
2 #smiles("CCO", show-all-h: true)
```

 Typst




4 Atoms and charges

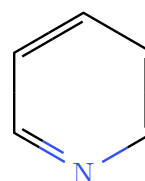
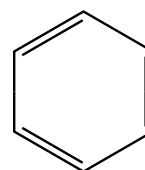
Standard SMILES syntax (atoms, bonds, branches, ring closures) is supported. This section covers the points specific to typed-smiles.

4.1 Aromatic rings

Note. The bundled parser does not yet read lowercase aromatic atoms (c1ccccc1). Write aromatic rings in Kekulé form: uppercase atoms with explicit alternating double bonds.

```
1 #smiles("C1=CC=CC=C1") \  
2 #smiles("C1=CC=NC=C1")
```


 Typst

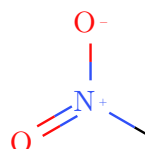
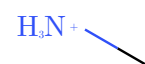
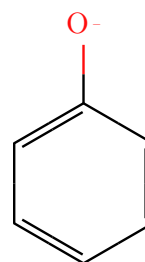


4.2 Charges

Formal charges inside brackets render as a raised sign after the atom.

```
1 #smiles("[O-]C1=CC=CC=C1") \  
2 #smiles("C[NH3+]") \  
3 #smiles("C[N+](=O)[O-]")
```

 Typst




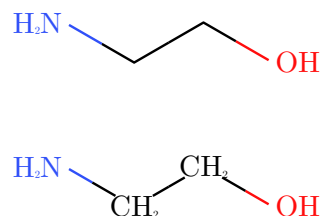
Note. A bracketed element such as [N] is a nitrogen atom. To print an arbitrary text label instead, use the abbreviation syntax {N} (see [Section 8](#)).

5 Hydrogens

5.1 Default display

Heteroatom hydrogens are shown by default; carbon hydrogens are hidden for a clean skeleton. `show-all-h: true` labels carbon hydrogens as well.


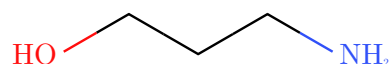
```
1 #smiles("OCCN") \  
2 #smiles("OCCN", show-all-h: true)
```

 Typst

5.2 Label orientation

Terminal heteroatom labels orient so the bond meets the heavy atom rather than the trailing hydrogen, at any angle.

```
1 #smiles("NCCCCO")
```

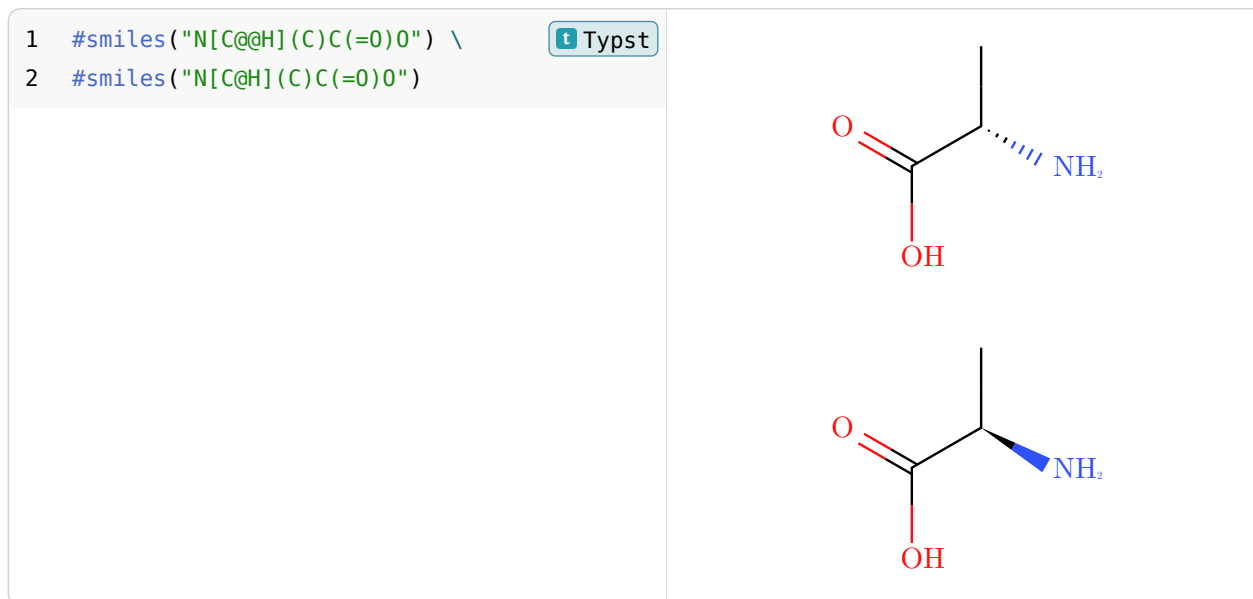
 Typst

6 Stereochemistry

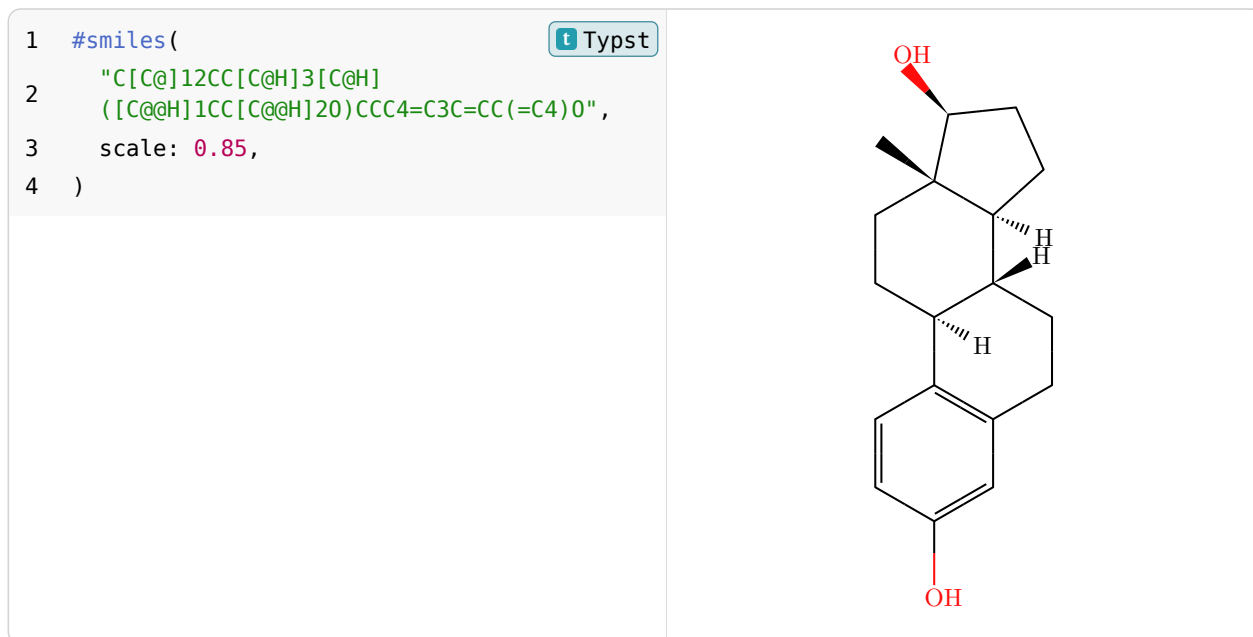
typed-smiles supports both standard SMILES stereo notations, plus a manual override for drawing wedges directly.

6.1 Tetrahedral centers: @ and @@

A bracket atom carrying @ or @@ becomes a wedge (toward the viewer) or hashed bond (away), computed from the 2D layout so the depiction matches the SMILES in the conventional orientation.



The renderer wedges an exocyclic substituent (such as OH or CH₃) where one exists, and an explicit hydrogen otherwise. Fused systems work too.




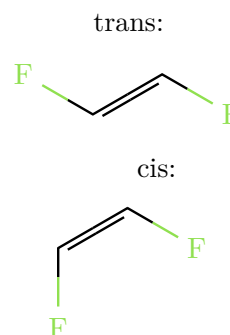
Note. The geometry is drawn correctly, but R/S descriptors are not computed. Ring stereochemistry between adjacent centers and bridged bicyclics may need a manual adjustment (see [Section 12](#)).

6.2 Double bonds: / and \

Directional bonds / and \ around a double bond set its cis/trans geometry. They come in pairs, one on each side of the =.

```
1 trans: #smiles("F/C=C/F")
2 #h(2em)
3 cis: #smiles("F/C=C\F")
```


 Typst

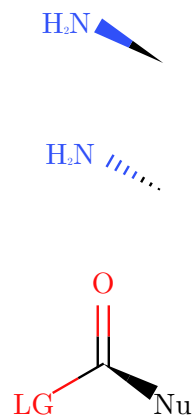


6.3 Manual wedges: !w and !h

For direct control, !w draws a solid wedge and !h a hashed wedge. These are typed-smiles extensions, not standard SMILES. Like plain bonds, they are colored by the atoms at each end.

```
1 #smiles("C!wN") \
2 #smiles("C!hN") \
3 #smiles("{Nu}!wC({LG|red})=O")
```









 Typst




7 Colors

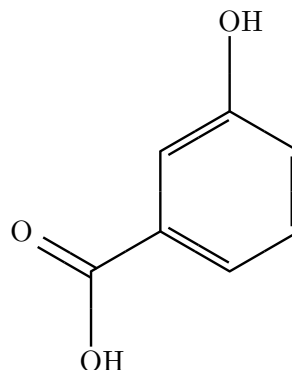
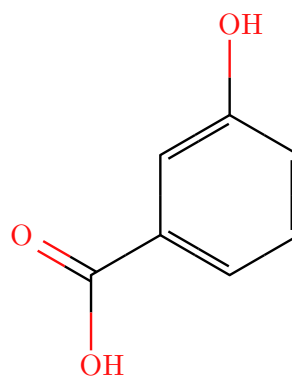
7.1 Jmol CPK palette

Atoms use the Jmol CPK palette by default. Carbon and unlisted elements are black; common heteroatoms get their conventional colors.

N	O	S	P	F	Cl	Br	I
							

Set `color: false` for a monochrome diagram.

```
1 #smiles("OC1=CC(=CC=C1)C(=O)O") \  Typst
2 #smiles("OC1=CC(=CC=C1)C(=O)O", color:
  false)
```




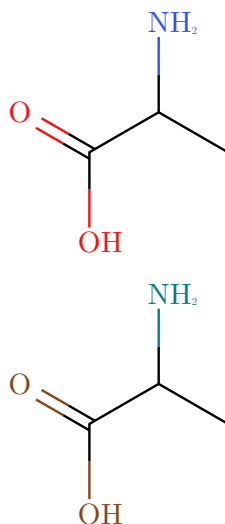
7.2 Custom colors (`atom-colors`)

`atom-colors` is a single dictionary that overrides colors for both elements and arbitrary abbreviated groups. Any Typst color value works — `rgb()`, `luma()`, named constants, or anything else.

Element-symbol keys

Use an element symbol as the key to replace that element's CPK color everywhere it appears, including when referenced as an inline label style (`{OMe|O}` would pick up the overridden oxygen color).

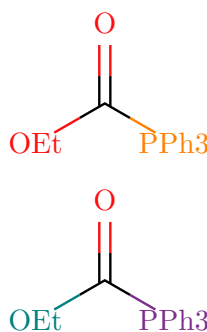
```
1 // default CPK
2 #smiles("CC(N)C(=O)O")
3 // oxygen → dark brown; nitrogen → teal
4 #smiles("CC(N)C(=O)O",
5   atom-colors: (O: rgb("#8B4513"), N: rgb("#008080")))
```

 Typst

Label-name keys

Wrap the label text in braces to target a specific abbreviated group, regardless of any inline style it carries. The key is written as a quoted string because `{` is not a valid Typst identifier character.


```
1 // default – colors come from inline styles or element fallback
2 #smiles("{PPh3|P}C({OEt|O})=O")
3 // override by label name – inline style is ignored for these
4 #smiles("{PPh3|P}C({OEt|O})=O",
5   atom-colors: ("{PPh3}": rgb("#7B2D8B"), "{OEt}": rgb("#008080")))
```

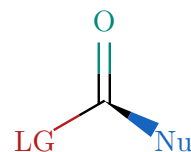
 Typst

Mixing both key forms

Element keys and label keys can coexist in the same dictionary.

```
// 0 (element) → teal; {Nu}
1 (label) → navy; {LG} (label) → maroon
2 #smiles("{Nu}!wC({LG|red})=O",
3   atom-colors: (0: rgb("#00897B"), "{Nu}":
   rgb("#1565C0"), "{LG}": rgb("#B71C1C"))
```

 Typst



Priority

Color is resolved in this order for each atom or label:

1. Label-name key in `atom-colors` (e.g. "{PPh3}"), if the atom is an abbreviation.
2. Element-symbol key in `atom-colors` (e.g. 0), for both real atoms and element-style labels.
3. The inline `{label|style}` style from the SMILES string (named color, hex, or element symbol).
4. The default CPK palette.

Note. `color: false` is a hard override that sits above all of the above. When it is set, every atom and label renders in black regardless of any `atom-colors` entries or inline styles. If you want a mostly-monochrome diagram with one or two highlighted groups, keep `color: true` (the default) and put everything else in `atom-colors`.

7.3 Label colors ({label|style})


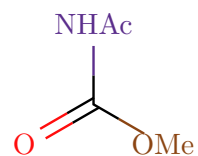
In the SMILES string, write `{label|style}` to color an abbreviated group independently of its element. The style can be a named color, an element symbol, or a hex code.

Named colors:

Name	Swatch	Name	Swatch
red		orange	
yellow		brown	
green		lime	
teal		cyan	
blue		navy	
purple		pink	
black		gray	
silver		maroon	
white			


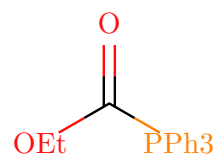
Hex colors: write a #RRGGBB hex code after the pipe to use any color. The # is just a regular character inside a Typst string literal.

```
1 #smiles("{OMe|#8B4513}C(=O){NHAc|#5B2E8C}")
```

 Typst

Element symbol: using a symbol as the style applies that element's (possibly overridden) CPK color to the label and its bonds.

```
1 #smiles("{PPh3|P}C({OEt|O})=O")
```


 Typst

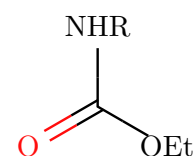
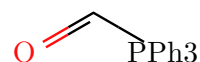
8 Abbreviated groups

8.1 Plain labels

Wrap text in braces `{...}` to place a labeled pseudo-atom that bonds like any other atom. Use it for groups you do not want to draw in full.

```
1 #smiles("{PPh3}C=O") \
2 #smiles("{OEt}C(=O){NHR}")
```


 Typst

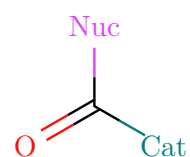
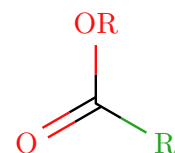
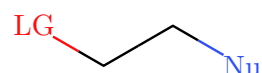


8.2 Colored labels

Add `|style` inside the braces to color a label. Use a named color, an element symbol, or a hex code — see [Section 7](#) for the full list.

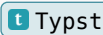
```
1 #smiles("{Nu|blue}CC{LG|red}") \
2 #smiles("{R|green}C(=O){OR|O}") \
3 #smiles("{Cat|teal}C(=O){Nuc|#E040FB}")
```

 Typst



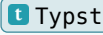
9 Chemical formulas: `ce()`

For formulas and text equations, `ce` is re-exported from the [chemformula](#) package, so one import covers both structures and formulas.

<pre>1 #ce("H2SO4") #h(1em) #ce("Ca^2+") #h(1em) #ce("2 H2 + O2 -> 2 H2O")</pre>	 Typst $\text{H}_2\text{SO}_4 \quad \text{Ca}^{2+} \quad 2\text{H}_2 + \text{O}_2 \longrightarrow 2\text{H}_2\text{O}$
---	---

9.1 Fonts and size

`ce` accepts `font` and `font-size` any other arguments pass through to `chemformula`.

<pre>1 #ce("CuSO4 * 5 H2O") \ 2 #ce("CuSO4 * 5 H2O", font: "Libertinus Serif", font-size: 13pt) \ 3 #ce("CuSO4 * 5 H2O", font: "PT Sans", font-size: 13pt)</pre>	 Typst $\begin{array}{l} \text{CuSO}_4 \cdot 5\text{H}_2\text{O} \\ \text{CuSO}_4 \cdot 5\text{H}_2\text{O} \\ \text{CuSO}_4 \cdot 5\text{H}_2\text{O} \end{array}$
--	--


10 Reaction schemes

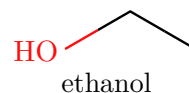
Three helpers compose molecules, formulas, and arrows into schemes.

10.1 mol()

`mol(content, label: none)` wraps any content and centers an optional caption beneath it.

```
1 #mol(smiles("CCO"), label:  
  [ethanol])
```

 Typst




10.2 rxn-arrow()

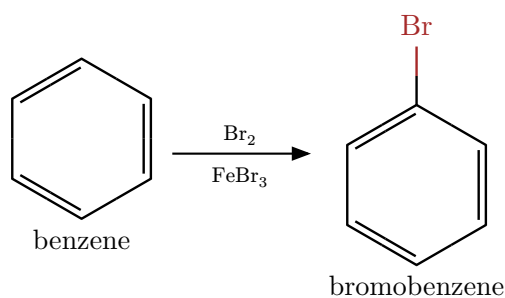
`rxn-arrow(above: none, below: none, dir: "right")` draws an arrow. `dir` may be "right", "left", "up", or "down". `above` and `below` carry reagents and conditions.

10.3 reaction()

`reaction(gap-h: 1.5em, gap-v: 1.5em, scale: 1.0, breakable: false, ..items)` lays out molecules and arrows left to right. An up or down arrow wraps the scheme onto a new row.

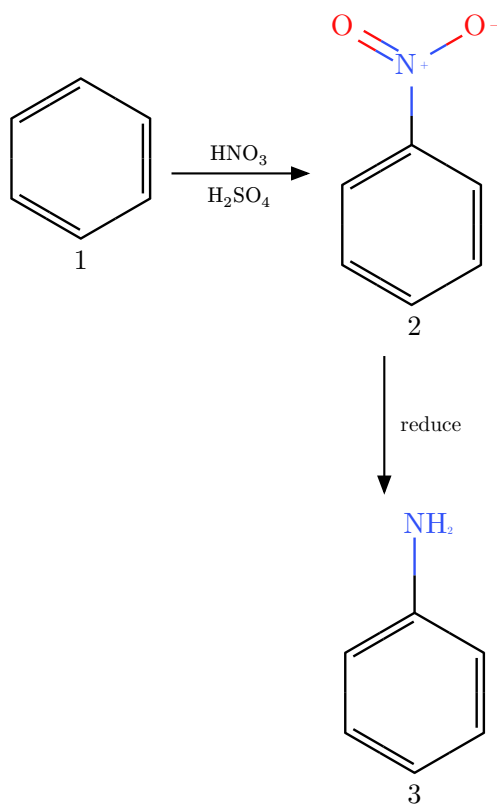
```
1 #reaction(  
2   mol(smiles("C1=CC=CC=C1"), label: [benzene]),  
3   rxn-arrow(above: ce("Br2"), below: ce("FeBr3")),  
4   mol(smiles("BrC1=CC=CC=C1"), label: [bromobenzene]),  
5 )
```

 Typst



A wrap-around scheme using a downward arrow:

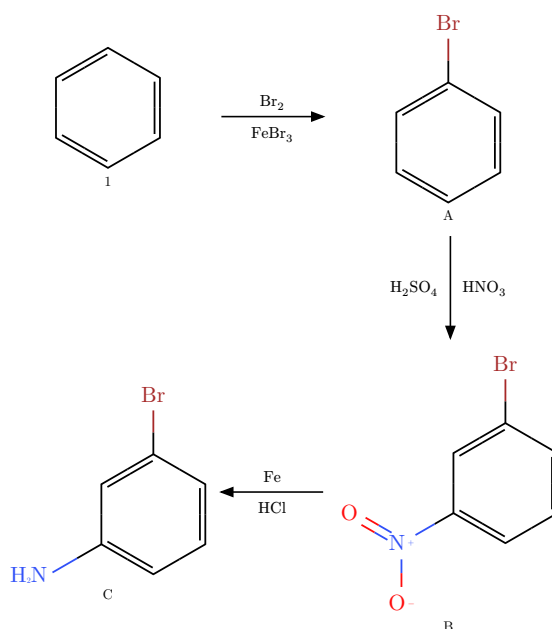
```
1 #reaction(  
2   mol(smiles("C1=CC=CC=C1"), label: [1]),  
3   rxn-arrow(above: ce("HNO3"), below: ce("H2SO4")),  
4   mol(smiles("[O-][N+](=O)C1=CC=CC=C1"), label: [2]),  
5   rxn-arrow(dir: "down", above: [reduce]),  
6   mol(smiles("NC1=CC=CC=C1"), label: [3]),  
7 )
```

[t Typst](#)

Uniform scale

Pass `scale` to shrink or enlarge the entire scheme uniformly — molecules, arrows, labels, and separators all change together. This is especially useful for multi-step or wrap-around schemes that would otherwise be too wide.

```
1 // same scheme at three different scales
2 #reaction(
3   scale: 0.75,
4   mol(smiles("C1=CC=CC=C1"), label: text(size: 7pt)[1]),
5   rxn-arrow(above: ce("Br2"), below: ce("FeBr3")),
6   mol(smiles("BrC1=CC=CC=C1"), label: text(size: 7pt)[A]),
7   rxn-arrow(dir: "down", above: ce("HNO3"), below: ce("H2SO4")),
8   mol(smiles("BrC1=CC(=CC=C1)[N+](=O)[O-]"), label: text(size: 7pt)[B]),
9   rxn-arrow(dir: "left", above: ce("Fe"), below: ce("HCl")),
10  mol(smiles("BrC1=CC(=CC=C1)N"), label: text(size: 7pt)[C]),
11 )
```

 Typst

Note. `reaction(scale: ...)` and `smiles(scale: ...)` are independent. The reaction scale is applied on top of however each individual molecule is sized. To resize everything from a single place, use `reaction(scale: ...)` and let each `smiles()` call use its default.

Page-break behaviour

By default, `reaction` sets `breakable: false`, so the whole scheme moves to the next page as a unit if it does not fit on the current one. This prevents a molecule or vertical branch from stranding on a different page from the rest of the scheme. Set `breakable: true` only for very long schemes that must span pages.

```
1 // Default – the scheme always stays on one page:
2 #reaction( ... )           // breakable: false
3
4 // Opt in to page splitting for very long schemes:
5 #reaction(breakable: true, ... )
```




11 Project-wide defaults

Typst functions support partial application with `.with()`. Calling `smiles.with(...)` returns a new function with the given arguments pre-filled. Rebind the name in your preamble and every subsequent `#smiles(...)` call uses those defaults — including inside `#reaction()`, because the content is evaluated before `reaction` sees it.

Any argument can be pre-filled this way: `bond-length`, `font`, `scale`, `font-size`, `atom-colors`, or any other `smiles` parameter.

```
1 // — preamble —————
2 #import "@preview/typed-smiles:0.3.0": *
3
4 #let smiles = smiles.with(
5   bond-length: 0.9,
6   font:       "Libertinus Serif",
7   atom-colors: (O: rgb("#8B4513"), N: rgb("#008080")),
8 )
9
10 // — anywhere in the document —————
11 #smiles("CC(N)C(=O)O") // uses bond-length 0.9, Libertinus, custom O/N
12 #reaction(
13   smiles("CCO"),
14   rxn-arrow(),
15   smiles("CC(=O)O"), // same custom smiles – preamble applies here too
16 )
```

 Typst

Note. Any argument passed directly at the call site still overrides the `.with()` default for that call alone.

12 Limitations

- Lowercase aromatic SMILES (c1ccccc1) are not parsed. Use Kekulé notation.
- Directional bonds (*/*, **) draw the correct cis/trans geometry, but E/Z descriptors are not computed.
- R/S and E/Z descriptors are not calculated.
- Ring stereochemistry between adjacent centers and bridged bicyclics can overlap or need a manual adjustment (try `rotation`, or the manual `!w` and `!h` wedges).

13 Quick reference

13.1 Syntax

Syntax	Meaning
[...]	Bracket atom (any element, charges, explicit H).
@ / @@	Tetrahedral anticlockwise / clockwise.
/ \	Double-bond cis/trans geometry.
!w / !h	Manual solid / hashed wedge (typed-smiles extension).
{label}	Abbreviated group pseudo-atom.
{label style}	Colored abbreviated group.

13.2 smiles() options







Option	Default	Effect
scale	1.0	Balanced size of everything.
bond-length	none	Bond length only (1.0 is 30 pt).
font-size	none	Atom-label size only.
font	"New Computer Modern"	Atom-label font.
bond-stroke	none	Bond width only.
color	true	CPK colors on or off.
rotation	0deg	Rotate, labels stay upright.
show-all-h	false	Label carbon hydrogens too.
atom-colors	(:)	Color overrides: 0: red for elements, "{PPh3}": blue for labels.










13.3 reaction() options

Option	Default	Effect
gap-h	1.5em	Horizontal gap between items.
gap-v	1.5em	Vertical gap between rows.
scale	1.0	Uniform scale applied to the whole scheme.
breakable	false	Allow the scheme to split across pages.

13.4 Label color names

Named styles for {label|style} — any of these plus any #RRGGBB hex:

Name	Swatch	Name	Swatch	Name	Swatch
red		orange		yellow	
brown		green		lime	

teal		cyan		blue	
navy		purple		pink	
black		gray		silver	
maroon		white	