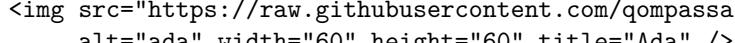
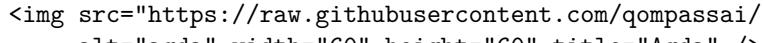
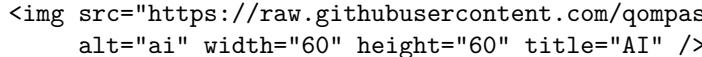


## Contents

OR . . . . .	13
OR . . . . .	31
<b>OR</b>	<b>37</b>
<b>OR</b>	<b>37</b>
<b>OR</b>	<b>42</b>
 Ada	
ada_ls	
Ada LSP Reference	
<code>mkdir -p ~/.local/src cd ~/.local/src wget https://github.com/alire-project/alire/releases/download/v2.0.2/alr-2.0.2-bin-x86_64-linux unzip alr-2.0.2-bin-x86_64-linux.zip cp bin/alr ~/.local/bin/ mkdir -p ~/.config/alire cat &gt; ~/.config/alire/config.toml &lt;&lt; 'EOF' [settings] cache_dir = "\$XDG_CACHE_HOME/alire" config_dir = "\$XDG_CONFIG_HOME/alire" toolchain_dir = "\$XDG_DATA_HOME/alire/toolchains" EOF alr toolchain --select gnat_native alr toolchain --select gprbuild mkdir -p ~/tmp/als-install &amp;&amp; cd ~/tmp/als-install alr init --bin als_installer cd als_installer alr with ada_language_server alr build --release cp -v ~/.local/share/alire/builds/ada_language_server*/* ~/local/bin/</code>	
 Agda	
agda_ls	
Agda LSP Reference	

```

git clone https://github.com/banacorn/agda-language-server && cd agda-langauge-server \
git submodule update --init --recursive && stack install


</div>
<strong>AI</strong>

ai_ls

<p>
AI LSP Reference
</p>

cargo install lsp-ai -F llama_cpp -F cuda

copilot_ls

Copilot LSP Reference

pnpm add -g @github/copilot-language-server@latest

text_ls

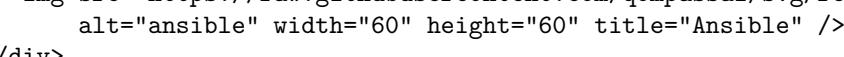
<p>
Text LSP Reference
</p>

pip install git+https://github.com/PrithivirajDamodaran/Gramformer.git

ai_ls

<p>
VectorCode LSP Reference
</p>

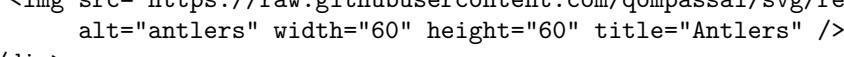
pip install "VectorCode[all]"


</div>
<strong>Ansible</strong>

ansible_ls

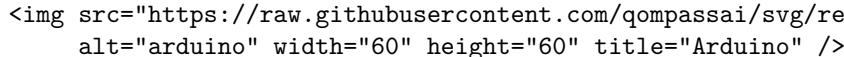
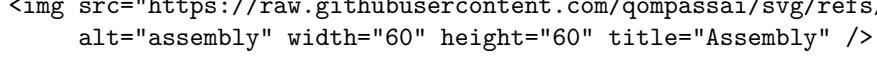
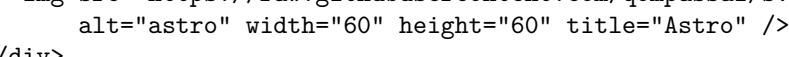
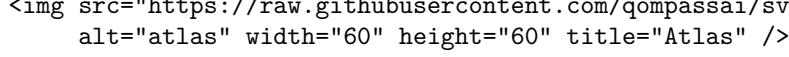
Ansible LSP Reference

pnpm add -g @ansible/ansible-language-server


</div>
<strong>Antlers</strong>

```

```

antlers_ls
Antlers LSP Reference
pnpm add -g antlers-language-server@latest

<strong>Arduino</strong>
arduino_ls
Arduino LSP Reference
go install github.com/arduino/arduino-language-server@latest

<strong>Assembly</strong>
asm_ls
<p>
  <a href="https://github.com/bergercookie/asm-lsp">Assembly LSP Reference</a>
</p>
cargo install --git https://github.com/bergercookie/asm-lsp asm-lsp
m68k_ls
<p>
  <a href="https://github.com/grahambates/m68k-lsp">M68K Assembly LSP Reference</a>
</p>
pnpm add -g m68k-lsp-server@latest

<strong>Astro</strong>
astro_ls
AstroJS LSP Reference
pnpm add -g typescript@latest @astrojs/language-server@latest prettier@latest prettier-plugin-ast

<strong>Atlas</strong>

```

```

atlas_ls
Atlas LSP Reference
curl -sSf https://atlasgo.sh | sh
</details>

</div>
<strong>Awk</strong>

awk_ls
<p>
  <a href="https://github.com/Beaglefoot/awk-language-server/">Awk LSP Reference</a>
</p>
pnpm add -g awk-language-server@latest
<details>

</div>
<strong>Azure Pipelines</strong>

azurepipelines_ls
Azure Pipelines LSP Reference
pnpm add -g azure-pipelines-language-server@latest

</div>
<strong>Bazelrc</strong>

bazelrc_ls
<p>
  <a href="https://www.npmjs.com/package/azure-pipelines-language-service/">Bazelrc LSP Reference</a>
</p>
pnpm add -g git+https://github.com/salesforce-misc/bazelrc-lsp.git

</div>
<strong>Beancount</strong>

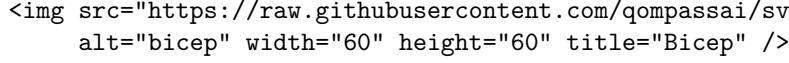
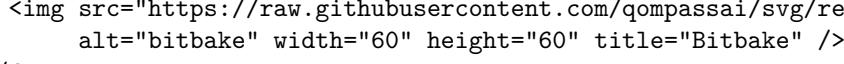
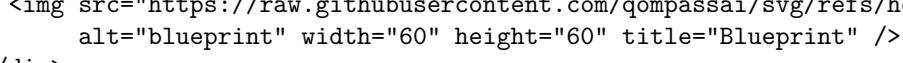
beancount_ls

```

```

<p>
  <a href="https://github.com/polarmutex/beancount-language-server#installation">Beancount LSP Reference</a>
</p>

cargo install beancount-language-server
OR
pip install beancount


</div>
<strong>Bicep</strong>
bicep_ls
<p>
  <a href="https://github.com/azure/bicep">Bicep LSP Reference</a>
</p>
: "${XDG_DATA_HOME:=${HOME}/.local/share}""
(
  cd "$(mktemp -d)" \
  && curl -fLo https://github.com/Azure/bicep/releases/latest/download/bicep-langserver.zip \
  && rm -rf "$XDG_DATA_HOME/bicep-langserver" \
  && unzip -d "$XDG_DATA_HOME/bicep-langserver" bicep-langserver.zip
)

</div>
<strong>Bitbake</strong>
bitbake_ls
BitBake LSP Reference
pip install bitbake-language-server
OR
uv tool install bitbake-language-server

</div>
<strong>Blueprint</strong>
blueprint_ls

```

```

<p>
  <a href="https://gnome.pages.gitlab.gnome.org/blueprint-compiler/index.html">BluePrint Reference</a>
</p>

git clone https://gitlab.gnome.org/GNOME/blueprint-compiler --recursive \
&& cd blueprint-compiler && git fetch --all && git submodule update --init --recursive \
meson _build && ninja -C _build install


</div>
<strong>BQN</strong>

bq_ls
BigQuery LSP Reference

go install github.com/kitagry/bqls@latest


</div>
<strong>BrighterScript</strong>

bsc_ls
<p>
  <a href="https://github.com/RokuCommunity/brighterscript">BrighterScript Reference</a>
</p>

pnpm add -g brighterscript@latest


</div>
<strong>Brioche</strong>

brioche_ls
<p>
  <a href="https://brioche.dev/">Brioche Reference</a>
</p>

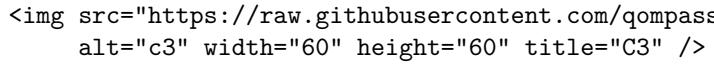
curl --proto '=https' --tlsv1.2 -sSfL 'https://brioche.dev/install.sh' | sh

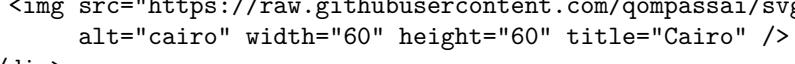
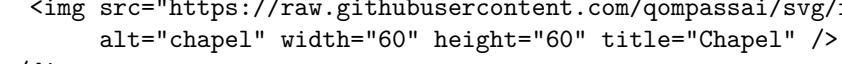

</div>
<strong>Buck2</strong>

buck2_ls
<p>
  <a href="https://buck2.build/docs/users/commands/lsp/#buck2-lsp">Buck2 LSP Reference</a>
</p>

```

```

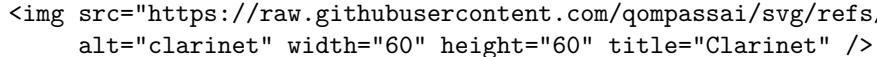
</p>
rustup install nightly-2025-08-01 && cargo +nightly-2025-08-01 install --git https://github.com/qompassai/c3-lsp

</div>
<strong>C3</strong>
c3_ls
C3 LSP Reference
git clone https://github.com/c3lang/c3-lsp.git --recursive \
cd c3-lsp && git fetch --all && git submodule update --init --recursive \
./scripts/build_linux.sh \
mv server/bin/c3lsp ~/.local/bin


</div>
<strong>Cairo</strong>
cairo_ls
Cairo LSP Reference
curl --proto '=https' --tlsv1.2 -sSf https://sh.starkup.dev | sh
OR
cargo install --git https://github.com/software-mansion/scarb scarb

</div>
<strong>Chapel</strong>
chpl_ls
Chapel LSP Reference
export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
export CHPL_PREFIX="$XDG_DATA_HOME/chapel"
git clone https://github.com/chapel-lang/chapel.git "$CHPL_PREFIX-src"
cd "$CHPL_PREFIX-src"
./configure --prefix="$CHPL_PREFIX"
share/chapel/x.yz
make -j$(nproc)
make install
share/chapel/x.yz
export CHPL_HOME="$CHPL_PREFIX/share/chapel/$(ls "$CHPL_PREFIX/share/chapel")"
export PATH="$CHPL_HOME/bin:$PATH"

```

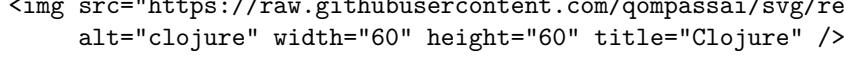
```

cd "$CHPL_HOME"
make chpl-language-server


<div>
<strong>Clarinet</strong>
clarinet_ls

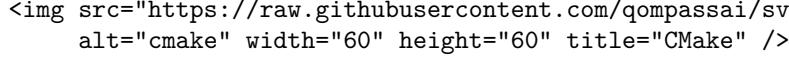
<p>
<a href=" https://github.com/hirosystems/clarinet">Clarinet Reference</a>
</p>

cargo install --git https://github.com/stx-labs/clarinet clarinet-cli


<div>
<strong>Clojure</strong>
clojure_ls

<p>
<a href="https://closure-lsp.io/">Closure LSP Reference</a>
</p>

export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
export CLOJURE_LSP_DIR="$XDG_DATA_HOME/closure-lsp"
mkdir -p "$CLOJURE_LSP_DIR"
curl -O https://raw.githubusercontent.com/closure-lsp/closure-lsp/master/install
chmod a+x install
./install --version nightly --dir "$CLOJURE_LSP_DIR"
mkdir -p "$HOME/.local/bin"
ln -s "$CLOJURE_LSP_DIR/bin/closure-lsp" "$HOME/.local/bin/closure-lsp"


<div>
<strong>CMake</strong>
cmake_ls

CMake LSP Reference

pip install cmake-language-server
OR

uv tool install cmake-language-server
neocmake_ls

NeoCmake LSP Reference

```

```

cargo install neocmakelsp --git https://github.com/neocmakelsp/neocmakelsp
    <details>
        
    </div>
    <strong>Core Data Services (CDS)</strong>
    cds_ls
    CDS LSP Reference
    pnpm add -g @sap/cds-lsp@latest
        
    </div>
    <strong>Crystal</strong>
    crystalline_ls
    Crystalline LSP Reference
    git clone https://github.com/elbywan/crystalline --recursive && cd crystalline \
shards install && mkdir bin \
crystal build ./src/crystalline.cr -o ./bin/crystalline --release --no-debug --progress -D
        
    </div>
    <strong>Cobol</strong>
    cobol_ls
    <p>
        <a href="https://github.com/eclipse-che4z/che-che4z-lsp-for-cobol">Cobol LSP Reference</a>
    </p>
    export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}" && \
mkdir -p "$XDG_DATA_HOME/cobol-lsp" && \
git clone https://github.com/eclipse-che4z/che-che4z-lsp-for-cobol.git --recursive && \
cd che-che4z-lsp-for-cobol && \
./BUILD.sh && \
cp clients/cobol-lsp-vscode-extension/jar/server.jar "$XDG_DATA_HOME/cobol-lsp/server.jar" & \
mkdir -p "$HOME/.local/bin" && \
cat > "$HOME/.local/bin/cobol-language-support" <<'EOF'
#!/usr/bin/env bash
XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
exec java -jar "$XDG_DATA_HOME/cobol-lsp/server.jar" "$@"
EOF
chmod +x "$HOME/.local/bin/cobol-language-support"

```

```


</div>
<strong>CSS</strong>
css_ls
CSS LSP Reference
pnpm add -g vscode-langservers-extracted@latest
cssmodule_ls
<p>
  <a href="https://github.com/antonk52/cssmodules-language-server">CSS Module LSP Reference</a>
</p>
pnpm add -g cssmodules-language-server@latest
tailwindcss_ls
Tailwind CSS LSP Reference
pnpm add -g @tailwindcss/language-server@latest

</div>
<strong>C#</strong>
csharp_ls
<p>
  <a href="https://github.com/razzmatazz/csharp-language-server">C# LSP Reference</a>
</p>
dotnet tool install --global csharp-ls
</ul>

</div>
<strong>Composition</strong>
codebook_ls
CodeBook LSP Reference
cargo install --git https://github.com/blopker/codebook codebook-lsp
harper_ls
<p>
  <a href="https://writewithharper.com/docs/about">Harper LSP Reference</a>
</p>

```

```

cargo install --git https://github.com/automattic/harper harper-ls
prosemd_ls

<p>
  <a href="https://github.com/kitten/prosemd-lsp">ProseMD LSP Reference</a>
</p>

cargo install --git https://github.com/kitten/prosemd-lsp
typos_ls

<p>
  <a href="https://github.com/tekumara/typos-lsp">Typos LSP Reference</a>
</p>

cargo install --git https://github.com/tekumara/typos-lsp
  
</div>
<strong>OpenTofu</strong>

tofu_ls

<p>
  <a href="https://github.com/opentofu/tofu-ls">OpenTofu LSP Reference</a>
</p>

go install github.com/opentofu/tofu-ls@latest
  
</div>
<strong>Deno</strong>

deno_ls

  
</div>
<strong>Docker</strong>

docker_ls

<p>
  <a href="https://github.com/rcjsuen/dockerfile-language-server-nodejs">Docker LSP Reference</a>
</p>

pnpm add -g dockerfile-language-server-nodejs@latest
dockercompose_ls

<p>
  <a href="https://github.com/nikeeee/dot-language-server">Docker Compose LSP Reference</a>

```

```

</p>
pnpm add -g @microsoft/compose-language-service@latest
</ul>

</div>
<strong>Dot</strong>
dot_ls
<p>
<a href="https://github.com/nikeeee/dot-language-server">Dot LSP Reference</a>
</p>
pnpm add -g dot-language-server@latest

</div>
<strong>DotNET</strong>
omnisharp_ls
<p>
<a href=" https://github.com/Omnisharp/omnisharp-roslyn
">Omnisharp LSP Reference
export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
OMNI_RID="${OMNI_RID:-linux-x64}"
OMNI_VERSION="$(
curl -fsSLI -o /dev/null -w '%{url_effective}' \
https://github.com/OmniSharp/omnisharp-roslyn/releases/latest |
sed 's#.*##'
)"
OMNI_DIR="$OMNI_ROOT/$OMNI_VERSION-$OMNI_RID"
mkdir -p "$XDG_DATA_HOME/dotnet/omnisharp"
cd "$OMNI_DIR"
ARCHIVE_NAME="omnisharp-$OMNI_RID.tar.gz"
ARCHIVE_URL="https://github.com/OmniSharp/omnisharp-roslyn/releases/download/$OMNI_VERSION/$
curl -fL "$ARCHIVE_URL" -o "$ARCHIVE_NAME"
tar -xzf "$ARCHIVE_NAME"
rm "$ARCHIVE_NAME"
[ -f run ] && chmod +x run
[ -f OmniSharp ] && chmod +x OmniSharp
roslyn_ls
<p>
```

```

<a href="https://github.com/dotnet/roslyn">Roslyn LSP Reference</a>
</p>

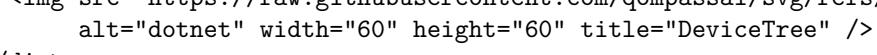
curl https://dot.net/v1/dotnet-install.sh -o /tmp/dotnet-install.sh \
chmod +x /tmp/dotnet-install.sh \
export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}" \
export DOTNET_ROOT="$XDG_DATA_HOME/dotnet" \
mkdir -p "$DOTNET_ROOT" \
/tmp/dotnet-install.sh --install-dir "$DOTNET_ROOT" --channel current \
export DOTNET_ROOT
export PATH="$DOTNET_ROOT:$PATH"
ROSLYN_LS_DIR="${XDG_DATA_HOME:-$HOME/.local/share}/dotnet/roslyn" \
mkdir -p "$ROSLYN_LS_DIR" \
cd "$ROSLYN_LS_DIR"
nuget install Microsoft.CodeAnalysis.LanguageServer -OutputDirectory "$ROSLYN_LS_DIR"

```

## OR

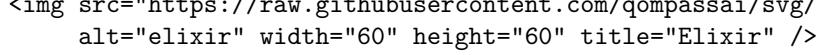
```

export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
ROSLYN_LS_DIR="$XDG_DATA_HOME/dotnet/roslyn"
mkdir -p "$ROSLYN_LS_DIR"
cd "$ROSLYN_LS_DIR"
nuget install Microsoft.CodeAnalysis.LanguageServer \
-OutputDirectory "$ROSLYN_LS_DIR"


<div>
<strong>Device Tree</strong>
dts_ls

<p>
<a href="https://github.com/igor-prusov/dts-lsp">DTS LSP Reference</a>
</p>

cargo install --git https://github.com/igor-prusov/dts-lsp


<div>
<strong>Elixir</strong>
elixir_ls

<p>
<a href="https://github.com/elixir-lsp/elixir-ls.git">Elixir LSP Reference</a>
</p>

```

```

curl https://elixir-lang.org/install.sh -o /tmp/elixir-install.sh
sh /tmp/elixir-install.sh \
  -y \
  --install-dir "${XDG_DATA_HOME:-$HOME/.local/share}/elixir-install"
expert_ls
Expert LSP Reference
git clone https://github.com/elixir-expert/expert.git \
  "${XDG_DATA_HOME:-$HOME/.local/share}/expert/src"
cd "${XDG_DATA_HOME:-$HOME/.local/share}/expert/src"
mix deps.get
mix compile
MIX_ENV=prod mix release \
  --path "${XDG_DATA_HOME:-$HOME/.local/share}/expert/release"


```

</div>

```

<strong>Elm</strong>
elm_ls
Elm LSP Reference
pnpm add -g elm@latest elm-test@latest elm-format@latest elm-review@latest @elm-tooling/elm


```

</div>

```

<strong>F*</strong>
fstar_ls
F* LSP Reference
opam install fstar


```

</div>

```

<strong>Fennel</strong>
fennel_ls
Fennel LSP Reference
luarocks --lua-version=5.1 install fennel-ls


```

</div>

```

<strong>F#</strong>

```

```

fsautocomplete_ls

<p>
<a href="https://github.com/fsharp/FsAutoComplete">F# AutoComplete LSP Reference</a>
</p>

dotnet tool install --global fsautocomplete


</div>
<strong>Git</strong>

ghactions_ls

Github Actions LSP Reference

pnpm add-g gh-actions-language-server@latest @actions/languageserver@latest
gitlabci_ls

<p>
<a href="https://github.com/alesbrelih/gitlab-ci-ls">Gitlab CI LSP Reference</a>
</p>

cargo install --git https://github.com/alesbrelih/gitlab-ci-ls.git


</div>
<strong>Gleam</strong>

gleam_ls

Gleam LSP Reference

XDG_DATA_HOME="${XDG_DATA_HOME:-"$HOME/.local/share"}"
XDG_BIN_HOME="${XDG_BIN_HOME:-"$HOME/.local/bin"}"
mkdir -p "$XDG_DATA_HOME" "$XDG_BIN_HOME"
cd "$XDG_DATA_HOME"
git clone https://github.com/gleam-lang/gleam.git --branch "$THE_LATEST_VERSION"
cd gleam
export CARGO_HOME="${XDG_DATA_HOME}/cargo"
mkdir -p "$CARGO_HOME"
make install PREFIX="$HOME/.local"

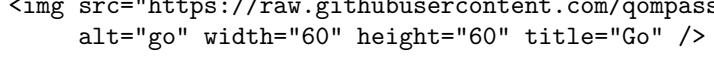

</div>
<strong>GLSL</strong>

glslana_ls

```

```

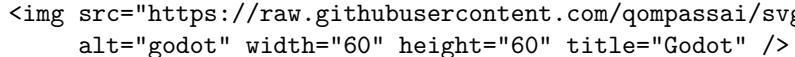
<p>
  <a href="https://github.com/nolanderc/glsl_analyzer">GLSL Analyzer LSP Reference</a>
</p>

XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"
mkdir -p "$XDG_DATA_HOME" "$XDG_BIN_HOME"
export ZVM_INSTALL="$XDG_DATA_HOME/zvm"
export PATH="$ZVM_INSTALL/bin:$PATH"
zvm upgrade
zvm install 0.14.0
cd "$XDG_DATA_HOME"
git clone https://github.com/nolanderc/glsl_analyzer.git --recursive
cd glsl_analyzer
zvm run 0.14.0 build install -Doptimize=ReleaseSafe --prefix "$HOME/.local"

</div>
<strong>Go</strong>
golangcilint_ls

<p>
  <a href="https://github.com/nametake/golangci-lint-langserver">Golang CI Lint LSP Reference</a>
</p>

go install github.com/golangci/golangci-lint/cmd/golangci-lint@latest && go install github.com/nametake/golangci-lint-langserver@latest
gop_ls

GoPls LSP Reference

go install golang.org/x/tools/gopl@latest

</div>
<strong>Godot</strong>
gdscript_ls

<p>
  <a href="https://github.com/godotengine/godot">Gdscript LSP Reference</a>
</p>

XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"
mkdir -p "$XDG_DATA_HOME/godot" "$XDG_BIN_HOME"
cd "$XDG_DATA_HOME/godot"
curl -L "https://downloads.godotengine.org/?version=4.5.1&flavor=stable&slug=linux.x86_64.zip"
  -o godot-4.5.1-linux.x86_64.zip

```

```

unzip godot-4.5.1-linux.x86_64.zip
rm godot-4.5.1-linux.x86_64.zip
chmod +x Godot_v4.5.1-stable_linux.x86_64
mv Godot_v4.5.1-stable_linux.x86_64 godot
cat > "$XDG_BIN_HOME/godot" <<'EOF'
#!/usr/bin/env bash
XDG_DATA_HOME="${XDG_DATA_HOME:-"$HOME/.local/share"}"
exec "$XDG_DATA_HOME/godot/godot" --single-window "$@"
EOF
chmod +x "$XDG_BIN_HOME/godot"
mkdir -p "$HOME/.local/share/applications"
cat > "$HOME/.local/share/applications/godot.desktop" <<EOF
[Desktop Entry]
Name=Godot
Exec=$XDG_BIN_HOME/godot
Terminal=false
Type=Application
Icon=$XDG_DATA_HOME/godot/icon.png
Categories=Development;Game;
EOF
update-desktop-database "$HOME/.local/share/applications" 2>/dev/null || true
gdshader_ls
Gdscript LSP Reference
cargo install --git https://github.com/GodOfAvacyn/gdshader-lsp && luarocks --lua-version=5.

</div>
<strong>GraphQL</strong>
graphql_ls
Graphql LSP Reference
pnpm add -g graphql-language-service-cli@latest graphql@latest

</div>
<strong>Groovy</strong>
groovy_ls
<p>
    <a href="https://github.com/prominic/groovy-language-server.git">Groovy LSP Reference</a>
</p>

```

```

git clone https://github.com/GroovyLanguageServer/groovy-language-server.git --recursive
cd groovy-language-server
./gradlew build
XDG_DATA_HOME="${XDG_DATA_HOME:-"$HOME/.local/share"}"
XDG_BIN_HOME="${XDG_BIN_HOME:-"$HOME/.local/bin"}"
mkdir -p "$XDG_DATA_HOME/groovy-language-server" "$XDG_BIN_HOME"
cp build/libs/groovy-language-server-all.jar \
    "$XDG_DATA_HOME/groovy-language-server/groovy-language-server-all.jar"


</div>
<strong>Helm</strong>

helm_ls

<p>
    <a href="https://pkg.go.dev/github.com/mrjosh/helm-ls">Helm LSP Reference</a>
</p>

OS=linux
ARCH=amd64
XDG_BIN_HOME="${XDG_BIN_HOME:-"$HOME/.local/bin"}"
mkdir -p "$XDG_BIN_HOME"
curl -L "https://github.com/mrjosh/helm-ls/releases/download/master/helm_ls_${OS}_${ARCH}" \
    --output "${XDG_BIN_HOME}/helm_ls"
chmod +x "${XDG_BIN_HOME}/helm_ls"


</div>
<strong>Htmt&EmbeddedRuby(Herb)</strong>

herb_ls

Herb LSP Reference

pnpm add -g @herb-tools/language-server@latest


</div>
<strong>Hoon</strong>

hoon_ls

Hoon LSP Reference

pnpm add -g @curbit/hoon-language-server@latest

<details>

```

```


</div>
<strong>Hydra</strong>
hydra_ls

<p>
<a href="https://github.com/Retsediv/hydra-lsp">Hydra LSP Reference</a>
</p>

pip install hydra-lsp


</div>
<strong>Hyprland</strong>
hypr_ls

<p>
<a href="https://en.gwen.works/hyprls/">Hypr LSP Reference</a>
</p>

go install github.com/hyprland-community/hyprls/cmd/hyprls@latest


</div>
<strong>Idris2</strong>
idris2_ls

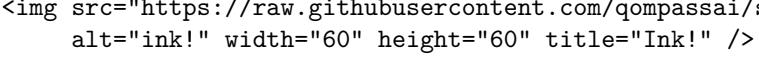
<p>
<a href="https://github.com/idris-community/idris2-lsp">Idris2 LSP Reference</a>
</p>

set -euo pipefail
export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
export IDRIS2_PREFIX="${IDRIS2_PREFIX:-$XDG_DATA_HOME/idris2}"
REPO_DIR="${REPO_DIR:-$HOME/.GH/idris2-lsp-src}"
mkdir -p "$IDRIS2_PREFIX" "$REPO_DIR"
if [ ! -d "$REPO_DIR/.git" ]; then
    git clone https://github.com/idris-community/idris2-lsp.git "$REPO_DIR"
else
    git -C "$REPO_DIR" pull --ff-only
fi
cd "$REPO_DIR"
git config protocol.https.allow always
git submodule update --init Idris2
cd Idris2

```

```

export SCHEME=chez
export IDRIS2_VERSION=0.8.0
make all \
PREFIX="$IDRIS2_PREFIX" IDRIS2_PREFIX="$IDRIS2_PREFIX"
make install \
PREFIX="$IDRIS2_PREFIX" IDRIS2_PREFIX="$IDRIS2_PREFIX"
make install-with-src-libs \
PREFIX="$IDRIS2_PREFIX" IDRIS2_PREFIX="$IDRIS2_PREFIX"
make install-with-src-api \
PREFIX="$IDRIS2_PREFIX" IDRIS2_PREFIX="$IDRIS2_PREFIX"
cd "$REPO_DIR"
case ":$PATH:" in
*:{$IDRIS2_PREFIX/bin}:*) ;;
*) export PATH="$IDRIS2_PREFIX/bin:$PATH" ;;
esac
git submodule update --init LSP-lib
cd LSP-lib
idris2 --install-with-src
cd "$REPO_DIR"
make install PREFIX="$IDRIS2_PREFIX" IDRIS2_PREFIX="$IDRIS2_PREFIX"
echo "idris2 & idris2-lsp installed to $IDRIS2_PREFIX/bin"
echo "Add this to your shell config (e.g. ~/.bashrc, ~/.zshrc):"
echo "  export XDG_DATA_HOME=\"\${XDG_DATA_HOME:-\$HOME/.local/share}\\""
echo "  export IDRIS2_PREFIX=\"\${IDRIS2_PREFIX:-\$XDG_DATA_HOME/idris2}\\""
echo "  export PATH=\"\$IDRIS2_PREFIX/bin:\$PATH\""



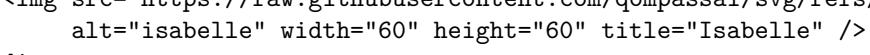
```

**Ink!**

ink\_ls

<p>
[Ink! LSP Reference](https://github.com/ink-analyzer/ink-analyzer/tree/master/crates/lsp-server)

cargo install --git https://github.com/ink-analyzer/ink-analyzer.git



</div>

**Isabelle**

isabelle\_ls

<p>
[Isabelle LSP Reference](https://github.com/ThreeFx/isabelle-lsp)
</p>

```

go install github.com/ThreeFx/isabelle-lsp@latest


</div>
<strong>Janet</strong>

janet_ls

<p>
    <a href="https://github.com/CFiggers/janet-lsp">Janet LSP Reference</a>
</p>

# Having janet and jpm already installed
git clone https://github.com/CFiggers/janet-lsp --recursive \
cd janet-lsp \
sudo jpm deps \
sudo jpm build \
sudo jpm install


</div>
<strong>Java</strong>

java_ls

Java LSP Reference

XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"
JLS_DIR="${XDG_DATA_HOME}/java-language-server"
BIN_DIR="${XDG_BIN_HOME}"
REPO_URL="https://github.com/georgewfraser/java-language-server.git"
mkdir -p "${JLS_DIR}" "${BIN_DIR}"
if [ ! -d "${JLS_DIR}/.git" ]; then
    git clone "${REPO_URL}" --recursive "${JLS_DIR}"
else
    cd "${JLS_DIR}"
    git pull --ff-only
fi
cd "${JLS_DIR}"
./scripts/download_linux.sh
./scripts/link_linux.sh
mvn package -DskipTests
WRAPPER="${BIN_DIR}/java-language-server"
cat > "${WRAPPER}" <<EOF
#!/usr/bin/env sh
exec "${JLS_DIR}/dist/lang_server_linux.sh" "\$@"
EOF
chmod +x "${WRAPPER}"

```

```

if [ -f "${HOME}/.bashrc" ]; then
    if ! grep -q 'XDG_BIN_HOME' "${HOME}/.bashrc"; then
    {
        printf '\n# Add XDG_BIN_HOME to PATH (java-language-server installer)\n'
        printf 'export XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"\n'
        printf 'export PATH="$XDG_BIN_HOME:$PATH"\n'
    } >> "${HOME}/.bashrc"
    echo "Updated ~/.bashrc to include XDG_BIN_HOME on PATH."
    fi
fi
FISH_CONFIG_DIR="${HOME}/.config/fish"
mkdir -p "${FISH_CONFIG_DIR}"
FISH_CONFIG="${FISH_CONFIG_DIR}/config.fish"
if [ -f "${FISH_CONFIG}" ]; then
    if ! grep -q 'XDG_BIN_HOME' "${FISH_CONFIG}"; then
    {
        printf '\n# Add XDG_BIN_HOME to PATH (java-language-server installer)\n'
        printf 'set -q XDG_BIN_HOME; or set -Ux XDG_BIN_HOME $HOME/.local/bin\n'
        printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
    } >> "${FISH_CONFIG}"
    echo "Updated ${FISH_CONFIG} to include XDG_BIN_HOME on PATH."
    fi
else
{
    printf '# Add XDG_BIN_HOME to PATH (java-language-server installer)\n'
    printf 'set -q XDG_BIN_HOME; or set -Ux XDG_BIN_HOME $HOME/.local/bin\n'
    printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
} > "${FISH_CONFIG}"
fi
jdt_ls
<p>
<a href=" https://github.com/uros-5/jinja-lsp">JDT LSP Reference</a>
</p>
XDG_DATA_HOME="${XDG_DATA_HOME:-"$HOME/.local/share"}"
XDG_BIN_HOME="${XDG_BIN_HOME:-"$HOME/.local/bin"}"
JDTLS_DIR="${XDG_DATA_HOME}/jdtls"
BIN_DIR="${XDG_BIN_HOME}"
JDTLS_URL="http://download.eclipse.org/jdtls/snapshots/jdt-language-server-latest.tar.gz"
mkdir -p "${JDTLS_DIR}" "${BIN_DIR}"
TMP_TAR=$(mktemp /tmp/jdtls.XXXXXX.tar.gz)
curl -sSfL "${JDTLS_URL}" -o "${TMP_TAR}"
rm -rf "${JDTLS_DIR:?}/*"
tar -xzf "${TMP_TAR}" -C "${JDTLS_DIR}"
rm -f "${TMP_TAR}"

```

```

LAUNCHER_JAR="$(printf '%s\n' "${JDTLS_DIR}"/plugins/org.eclipse.equinox.launcher_*.jar | head -n 1)"
if [ ! -f "${LAUNCHER_JAR}" ]; then
    echo "Could not find Equinox launcher jar in ${JDTLS_DIR}/plugins" >&2
    exit 1
fi
JDTLS_DATA_DIR="${XDG_DATA_HOME}/jdtls-workspaces"
mkdir -p "${JDTLS_DATA_DIR}"
WRAPPER="${BIN_DIR}/jdtls"
cat > "${WRAPPER}" <<EOF
#!/usr/bin/env sh
XDG_DATA_HOME="\${XDG_DATA_HOME:-\$HOME/.local/share}"
JDTLS_DIR="\${JDTLS_DIR}"
LAUNCHER_JAR="\${LAUNCHER_JAR}"
JDTLS_DATA_DIR="\${JDTLS_DATA_DIR:-\$JDTLS_DATA_DIR}"
JAVA_BIN="\${JAVA_BIN:-java}"
exec "\$JAVA_BIN" \\
    -Declipse.application=org.eclipse.jdt.ls.core.id1 \\
    -Dosgi.bundles.defaultStartLevel=4 \\
    -Declipse.product=org.eclipse.jdt.ls.core.product \\
    -Dlog.protocol=true \\
    -Dlog.level=ALL \\
    -Xms1G \\
    -Xmx2G \\
    --add-modules=ALL-SYSTEM \\
    --add-opens java.base/java.util=ALL-UNNAMED \\
    --add-opens java.base/java.lang=ALL-UNNAMED \\
    -jar "\$LAUNCHER_JAR" \\
    -configuration "\$JDTLS_DIR/config_linux" \\
    -data "\$JDTLS_DATA_DIR/\$PWD##*/"
EOF
chmod +x "${WRAPPER}"
if [ -f "\$HOME/.bashrc" ]; then
    if ! grep -q 'XDG_BIN_HOME' "\$HOME/.bashrc"; then
        {
            printf '\n# Add XDG_BIN_HOME to PATH (Eclipse JDT LS installer)\n'
            printf 'export XDG_BIN_HOME="\$XDG_BIN_HOME:-\$HOME/.local/bin"\n'
            printf 'export PATH="\$XDG_BIN_HOME:\$PATH"\n'
        } >> "\$HOME/.bashrc"
    fi
fi
FISH_CONFIG_DIR="\$HOME/.config/fish"
mkdir -p "${FISH_CONFIG_DIR}"
FISH_CONFIG="\${FISH_CONFIG_DIR}/config.fish"
if [ -f "\${FISH_CONFIG}" ]; then
    if ! grep -q 'XDG_BIN_HOME' "\${FISH_CONFIG}"; then
        {

```

```

printf '\n# Add XDG_BIN_HOME to PATH (Eclipse JDT LS installer)\n'
printf 'set -q XDG_BIN_HOME; or set -Ux XDG_BIN_HOME $HOME/.local/bin\n'
printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
} >> "${FISH_CONFIG}"
fi
else
{
    printf '# Add XDG_BIN_HOME to PATH (Eclipse JDT LS installer)\n'
    printf 'set -q XDG_BIN_HOME; or set -Ux XDG_BIN_HOME $HOME/.local/bin\n'
    printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
} > "${FISH_CONFIG}"
fi


<strong>JavaScript</strong>

biome_ls
Biome LSP Reference
pnpm add -D -E @biomejs/biome@latest
eslint_ls
<p>
    <a href="https://github.com/danielpza/eslint-lsp">Eslint LSP Reference</a>
</p>
pnpm add -D -E eslint@latest
quicklint_js
QuickLintJS LSP Reference
pnpm add -D -E quick-lint-js@latest
markojs_ls
<p>
    <a href="https://github.com/marko-js/language-server">MarkoJS LSP Reference</a>
</p>
pnpm add -D -E @marko/language-server@latest
oxlint_ls
<p>
    <a href="https://www.npmjs.com/package/oxlint">Oxlint LSP Reference</a>
</p>
pnpm add -g oxlint@latest

```

```



```

</div>

<strong>Jimmer DTO</strong>

jimmerdto\_ls

```

<p>
    <a href="https://github.com/Enaium/jimmer-dto-lsp">Jimmer DTO Reference</a>
</p>

```

```

mkdir -p $XDG_DATA_HOME/jimmer-dto-lsp && \
curl -L https://github.com/Enaium/jimmer-dto-lsp/releases/latest/download/server.jar \
-o ~/.local/share/jimmer-dto-lsp/server.jar

```

```



```

</div>

<strong>Jinja</strong>

jinja\_ls

Jinja LSP Reference

```

cargo install jinja --git https://github.com/uros-5/jinja-lsp jinja-lsp

```

```



```

</div>

<strong>JQ</strong>

jq\_ls

JQ LSP Reference

```

go install github.com/wader/jq-lsp@latest

```

```



```

</div>

<strong>JSON\*</strong>

json\_ls

```

<p>
    <a href="https://www.npmjs.com/package/vscode-json-languageserver">JSON LSP Reference</a>
</p>

```

```

pnpm add -g vscode-json-languageserver@latest

```

#OR

```

npm i vscode-json-languageserver@latest

```

jsonld\_ls

```

<p>
  <a href="https://github.com/digitalbazaar/jsonld.js">Jsonnet LSP Reference</a>
</p>

pnpm add -g jsonld@latest

jsonnet_ls

<p>
  <a href="https://github.com/carlverge/jsonnet-lsp">Jsonnet LSP Reference</a>
</p>

go install github.com/carlverge/jsonnet-lsp@latest


</div>
<strong>Julia</strong>

julia_ls


</div>
<strong>Just</strong>

just_ls

Just LSP Reference

cargo install --git https://github.com/terror/just-lsp just-lsp


</div>
<strong>Kotlin</strong>

kotlin_ls

Kotlin LSP Reference

export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
git clone https://github.com/fwcd/kotlin-language-server.git --recursive
cd kotlin-language-server
./gradlew :server:installDist
KLS_DEST="$XDG_DATA_HOME/kotlin/kotlin-language-server"
mkdir -p "$KLS_DEST"
cp -a server/build/install/server/* "$KLS_DEST/"


</div>
<strong>LaTeX</strong>

```

ltxplus\_ls

Ltex+ LSP Reference

```
XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"
LSP_DATA_DIR="${XDG_DATA_HOME}/lsp-ltex-plus"
BIN_DIR="${XDG_BIN_HOME}"
REPO="ltxplus/ltex-ls-plus"
LATEST_TAG=$(curl -sSfL "https://api.github.com/repos/${REPO}/releases/latest" \
| grep -Eo '"tag_name":\s*"[^"]+"' \
| sed -E 's/.*(["]+").*/\1/'")
OS="linux"
ARCH="x64"
ARCHIVE_NAME="ltx-ls-plus-${LATEST_TAG}-${OS}-${ARCH}.tar.gz"
INSTALL_DIR="${LSP_DATA_DIR}/${LATEST_TAG}"
mkdir -p "${LSP_DATA_DIR}" "${BIN_DIR}"
curl -sSfL "https://github.com/${REPO}/releases/download/${LATEST_TAG}/${ARCHIVE_NAME}" \
-o "/tmp/${ARCHIVE_NAME}"
rm -rf "${INSTALL_DIR}"
mkdir -p "${INSTALL_DIR}"
tar -xzf "/tmp/${ARCHIVE_NAME}" -C "${INSTALL_DIR}"
if [ -x "${INSTALL_DIR}/bin/ltex-ls-plus" ]; then
    TARGET="${INSTALL_DIR}/bin/ltex-ls-plus"
elif [ -x "${INSTALL_DIR}/ltex-ls-plus" ]; then
    TARGET="${INSTALL_DIR}/ltex-ls-plus"
else
    echo "Could not find ltex-ls-plus binary in ${INSTALL_DIR}" >&2
    exit 1
fi
WRAPPER="${BIN_DIR}/ltex-ls-plus"
cat > "${WRAPPER}" <<EOF
#!/usr/bin/env sh
exec "${TARGET}" "\$@"
EOF
chmod +x "${WRAPPER}"
if [ -f "${HOME}/.bashrc" ]; then
    if ! grep -q 'XDG_BIN_HOME' "${HOME}/.bashrc"; then
        {
            printf '\n# Add XDG_BIN_HOME to PATH (Qompass lsp-ltex-plus installer)\n'
            printf 'export XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"\n'
            printf 'export PATH="$XDG_BIN_HOME:$PATH"\n'
        } >> "${HOME}/.bashrc"
    fi
fi
FISH_CONFIG_DIR="${HOME}/.config/fish"
mkdir -p "${FISH_CONFIG_DIR}"
```

```

FISH_CONFIG="${FISH_CONFIG_DIR}/config.fish"

if [ -f "${FISH_CONFIG}" ]; then
    if ! grep -q 'XDG_BIN_HOME' "${FISH_CONFIG}"; then
    {
        printf '\n# Add XDG_BIN_HOME to PATH (Qompass lsp-ltex-plus installer)\n'
        printf 'set -q XDG_BIN_HOME; or set -Ux XDG_BIN_HOME $HOME/.local/bin\n'
        printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
    } >> "${FISH_CONFIG}"
    fi
else
{
    printf '# Add XDG_BIN_HOME to PATH (Qompass lsp-ltex-plus installer)\n'
    printf 'set -q XDG_BIN_HOME; or set -Ux XDG_BIN_HOME $HOME/.local/bin\n'
    printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
} > "${FISH_CONFIG}"
fi

echo "ltex-ls-plus installed. Restart your shell to pick up PATH changes."
texlab_ls

<p>
<a href="https://github.com/latex-lsp/texlab">Texlab LSP Reference</a>
</p>

XDG_DATA_HOME="${XDG_DATA_HOME:-"$HOME/.local/share"}"
XDG_BIN_HOME="${XDG_BIN_HOME:-"$HOME/.local/bin"}"
TEXLAB_DATA_DIR="${XDG_DATA_HOME}/lsp-texlab"
BIN_DIR="${XDG_BIN_HOME}"
REPO="latex-lsp/texlab"
LATEST_TAG=$(curl -sSfL "https://api.github.com/repos/${REPO}/releases/latest" \
| grep -Eo '"tag_name":\s*[^"]+' \
| sed -E 's/.*"([^"]+").*/\1/'")
OS="linux"
ARCH="x86_64"
ARCHIVE_NAME="texlab-${OS}-${ARCH}.tar.gz"
INSTALL_DIR="${TEXLAB_DATA_DIR}/${LATEST_TAG}"
echo "Installing texlab ${LATEST_TAG} to ${INSTALL_DIR}"
mkdir -p "${TEXLAB_DATA_DIR}" "${BIN_DIR}"
curl -sSfL "https://github.com/${REPO}/releases/download/${LATEST_TAG}/${ARCHIVE_NAME}" \
-o "/tmp/${ARCHIVE_NAME}"
rm -rf "${INSTALL_DIR}"
mkdir -p "${INSTALL_DIR}"
tar -xzf "/tmp/${ARCHIVE_NAME}" -C "${INSTALL_DIR}"
if [ ! -x "${INSTALL_DIR}/texlab" ]; then
    echo "Could not find texlab binary in ${INSTALL_DIR}" >&2

```

```

    exit 1
fi
WRAPPER="${BIN_DIR}/texlab"
cat > "${WRAPPER}" <<EOF
#!/usr/bin/env sh
exec "${INSTALL_DIR}/texlab" "\$@"
EOF
chmod +x "${WRAPPER}"
if [ -f "${HOME}/.bashrc" ]; then
  if ! grep -q 'XDG_BIN_HOME' "${HOME}/.bashrc"; then
{
  printf '\n# Add XDG_BIN_HOME to PATH (Texlab installer)\n'
  printf 'export XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"\n'
  printf 'export PATH="$XDG_BIN_HOME:$PATH"\n'
} >> "${HOME}/.bashrc"
}
fi
FISH_CONFIG_DIR="${HOME}/.config/fish"
mkdir -p "${FISH_CONFIG_DIR}"
FISH_CONFIG="${FISH_CONFIG_DIR}/config.fish"
if [ -f "${FISH_CONFIG}" ]; then
  if ! grep -q 'XDG_BIN_HOME' "${FISH_CONFIG}"; then
{
  printf '\n# Add XDG_BIN_HOME to PATH (Texlab installer)\n'
  printf 'set -q XDG_BIN_HOME; or set -U XDG_BIN_HOME $HOME/.local/bin\n'
  printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
} >> "${FISH_CONFIG}"
}
else
{
  printf '# Add XDG_BIN_HOME to PATH (Texlab installer)\n'
  printf 'set -q XDG_BIN_HOME; or set -U XDG_BIN_HOME $HOME/.local/bin\n'
  printf 'set -U fish_user_paths $XDG_BIN_HOME $fish_user_paths\n'
} > "${FISH_CONFIG}"
fi
echo "texlab installed. Restart your shell to pick up PATH changes."

</div>
<strong>LLVM</strong>
mlir_ls
MLIR LSP Reference
git clone https://github.com/llvm/llvm-project.git --recursive \
cd llvm-project

```

```

mkdir build && cd build
cmake -G Ninja .. llvm \
-DLLVM_ENABLE_PROJECTS="mlir" \
-DCMAKE_BUILD_TYPE=Release \
-DCMAKE_INSTALL_PREFIX=$HOME/.local \
ninja mlir-lsp-server mlir-pdll-lsp-server \
ninja install

mlirpdll_ls
tblgen_ls


</div>
<strong>Logic</strong>

dolmen_ls
Dolmen LSP Reference
opam pin add https://github.com/Gbury/dolmen.git


</div>
<strong>Lua</strong>

Lua_ls
Lua LSP Reference
luarocks --lua-version=5.1 install lua-language-server
emmylua_ls

<p>
<a href="https://github.com/EmmyLuaLs/emmylua-analyzer-rust">Emmylua_ls LSP Reference</a>
</p>

cargo install --git https://github.com/EmmyLuaLs/emmylua-analyzer-rust schema_json_gen emmy
stylua_ls

<p>
<a href="https://github.com/JohnnyMorganz/StyLua">Stylua LSP Reference</a>
</p>

cargo install --git https://github.com/JohnnyMorganz/StyLua stylua --features luajit


</div>
<strong>Luau</strong>

```

```

Luau_ls

<p>
  <a href="https://luals.github.io/">luau_ls LSP Reference</a>
</p>

git clone https://github.com/JohnnyMorganz/luau-lsp.git --recurse-submodules \
cd luau-lsp && mkdir -p build && cd build \
cmake -S .. -B . \
-DCMAKE_BUILD_TYPE=Release \
-DCMAKE_CXX_FLAGS="-Wno-deprecated-literal-operator" \
-DCMAKE_INSTALL_PREFIX="$HOME/.local" && ninja \
cp luau-lsp ~/.local/bin


</div>
<strong>LWC</strong>

lwc_ls

LWC LSP Reference

pnpm add -g @salesforce/lwc-language-server@latest

</div>
<strong>Make</strong>

autotoo_ls

AutoTools LSP Reference

pip install autotools-language-server

```

## OR

```

uv tool install autotools-language-server

</div>
<strong>Markdown</strong>

moxide_ls

Moxide LSP Reference

cargo install --git https://github.com/Feel-ix-343/markdown-oxide
marksman_ls

Marksman LSP Reference

```

```

git clone https://github.com/artempyanykh/marksman.git --recursive && cd marksman
git fetch --all && git submodule update --init --recursive \
make install PREFIX="$HOME/.local"

mdxana_ls
Mdx Analyzer LSP Reference

pnpm add -g @mdx-js/typescript-plugin@latest
remark_ls
Remark LSP Reference

pnpm add -g remark-language-server@latest
Rumdl LSP Reference

cargo install --git https://github.com/rvben/rumdl


</div>
<strong>Matlab</strong>

matlab_ls
<p>
  <a href="https://github.com/mathworks/MATLAB-language-server">Matlab LSP Reference</a>
</p>

git clone https://github.com/mathworks/MATLAB-language-server.git && \
cd MATLAB-language-server && \
npm install && \
cd src/licensing/gui && npm install && cd ../../.. && \
npm run compile && \
mkdir -p ~/.local/share/matlab-language-server && \
cp -r out/* ~/.local/share/matlab-language-server/ && \
mkdir -p ~/.local/bin && \
cat > ~/.local/bin/matlab-language-server << 'EOF'
#!/bin/bash
exec node ~/.local/share/matlab-language-server/index.js "$@"
EOF
chmod +x ~/.local/bin/matlab-language-server && \
echo "MATLAB Language Server installed to ~/.local/bin/matlab-language-server"
Rumdl LSP Reference

cargo install --git https://github.com/rvben/rumdl


</div>

```

```

<strong>Mojo</strong>
mojo_ls

<p>
    <a href="https://github.com/modular/modular">Mojo Reference</a>
</p>

set -euo pipefail
: "${XDG_CONFIG_HOME:=$HOME/.config}"
: "${XDG_DATA_HOME:=$HOME/.local/share}"
: "${XDG_CACHE_HOME:=$HOME/.cache}"
: "${XDG_STATE_HOME:=$HOME/.local/state}"
export XDG_CONFIG_HOME XDG_DATA_HOME XDG_CACHE_HOME XDG_STATE_HOME
install_pixi() {
    if command -v pixi >/dev/null 2>&1; then
        echo "pixi already installed at: $(command -v pixi)"
        return
    fi
    if ! command -v cargo >/dev/null 2>&1; then
        echo "Error: cargo not found in PATH. Please install Rust/cargo first." >&2
        exit 1
    fi
    cargo install pixi
}
configure_pixi() {
    local pixi_config_dir="$XDG_CONFIG_HOME/pixi"
    local pixi_config_file="$pixi_config_dir/config.toml"

    mkdir -p "$pixi_config_dir"

    if [ ! -f "$pixi_config_file" ]; then
        cat > "$pixi_config_file" <<'EOF'
# Reference: https://prefix-dev.github.io/pixi/
default-channels = ["conda-forge"]
change-ps1 = true
tls-no-verify = false

[pypi-config]
index-url = "https://pypi.org/simple"
extra-index-urls = []
keyring-provider = "subprocess"
EOF
        echo "Created pixi config: $pixi_config_file"
    else
        echo "pixi config already exists: $pixi_config_file"
    fi
}

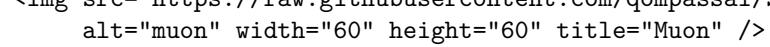
```

```

local pixi_manifest_dir="$XDG_CONFIG_HOME/pixi/manifests"
mkdir -p "$pixi_manifest_dir"
}
link_mojo_tools() {
    local mojo_env_dir="$XDG_DATA_HOME/mojo/.pixi/envs/default"
    local mojo_bin_dir="$mojo_env_dir/bin"
    local user_bin_dir="$HOME/.local/bin"
    mkdir -p "$user_bin_dir"
    if [ ! -x "$mojo_bin_dir/mojo-lsp-server" ]; then
        echo "Warning: $mojo_bin_dir/mojo-lsp-server not found or not executable." >&2
    else
        ln -sf "$mojo_bin_dir/mojo-lsp-server" "$user_bin_dir/mojo-lsp-server"
        echo "Linked mojo-lsp-server -> $user_bin_dir/mojo-lsp-server"
    fi
    if [ ! -x "$mojo_bin_dir/mojo-lldb-dap" ]; then
        echo "Warning: $mojo_bin_dir/mojo-lldb-dap not found or not executable." >&2
    else
        ln -sf "$mojo_bin_dir/mojo-lldb-dap" "$user_bin_dir/mojo-lldb-dap"
        echo "Linked mojo-lldb-dap -> $user_bin_dir/mojo-lldb-dap"
    fi
}
main() {
    install_pixi
    configure_pixi
    link_mojo_tools
}

main "$@"

```

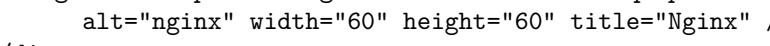
 Muon

muon\_ls

```

<p>
    <a href="https://muon.build">Muon Reference</a>
</p>
git clone https://github.com/muon-build/muon.git && cd muon && ./bootstrap.sh build \
&& build/muon-bootstrap setup build && build/muon-bootstrap -C build samu \
&& build/muon-bootstrap -C build test && sudo build/muon-bootstrap -C build install

```

 Nginx

```

nginx_ls

<p>
<a href="https://github.com/pappasam/nginx-language-server">Nginx LSP Reference</a>
</p>

pip install -U nginx-language-server


</div>
<strong>Nickel</strong>

nickel_ls

Nickel LSP Reference

cargo install --git https://github.com/tweag/nickel nickel-language-server


</div>
<strong>Nix</strong>

nil_ls

<p>
<a href="https://github.com/oxalica/nil">Nil LSP Reference</a>
</p>

cargo install --git https://github.com/oxalica/nil nil

#OR

nix profile install nixpkgs#nil

nixd_ls

Nixd LSP Reference

nix profile install github:nix-community/nixd

statix_ls


</div>
<strong>Nobl9</strong>

nobl9_ls

<p>
<a href="https://github.com/nobl9/nobl9-language-server">Nobl9 LSP Reference</a>
</p>

go install github.com/nobl9/nobl9-language-server/cmd/nobl9-language-server@latest

```

```


</div>
<strong>Ocaml</strong>
ocaml_ls


</div>
<strong>Odin</strong>
o_ls

Odin LSP Reference

git clone https://github.com/DanielGavin/ols.git --recursive \
cd ols && git fetch --all && git checkout dev-2025-11 \
./build.sh && ./odinfmt.sh \
mv ols ~/.local/bin \
mv odinfmt ~/.local/bin


</div>
<strong>OpenFOAM</strong>
foam_ls

<p>
<a href="https://github.com/FoamScience/foam-language-server">Foam LSP Reference</a>
</p>

pnpm add -g foam-language-server@latest


</div>
<strong>OpenPolicyAgent</strong>

regal_ls

<p>
<a href="https://github.com/StyraInc/regal">OPA Regal LSP Reference</a>
</p>

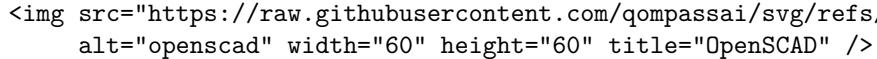
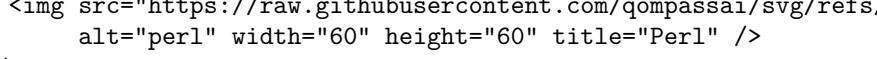
curl -L -o ~/.local/bin/regal \
"https://github.com/open-policy-agent/regal/releases/latest/download/regal_Linux_x86_64"

chmod +x ~/.local/bin/regal

rego_ls

```

```

</ul>
<p>
<a href="https://github.com/kitagry/regols">OPA Rego LSP Reference</a>
</p>
go install github.com/kitagry/regols@latest

</div>
<strong>OpenSCAD</strong>
openscad_ls
OpenSCAD LSP Reference
cargo install --git https://github.com/Leathong/openscad-LSP

</div>
<strong>Perl</strong>
perl_ls
Perl LSP Reference
cpanm Perl::LanguageServer

```

## OR

```

cpan Perl::LanguageServer
perlp_ls
PerlP LSP Reference
cpan PLS

```

## OR

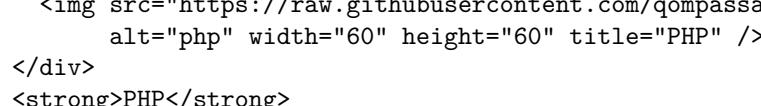
```

cpanm PLS
perlnav_ls
<p>
<a href="https://github.com/bscan/PerlNavigator/tree/main"> Perl Navigator Reference</a>
</p>
mkdir -p "$HOME/.local/bin"
cd "$(mktemp -d)"
curl -L -o perlnavigator.zip \

```

```

"https://github.com/bscan/PerlNavigator/releases/latest/download/perlnavigator-linux-x86_64.zip"
unzip perlnavigator.zip
cd perlnavigator-linux-x86_64
chmod +x perlnavigator
mv perlnavigator "$HOME/.local/bin/"


<strong>PHP</strong>

intelephense_ls
Intelephense LSP Reference
pnpm add -g intelephense@latest
phpactor_ls
PHPActor LSP Reference

PHP_BIN="${PHP_BIN:-php}"
INSTALL_DIR="${HOME}/.local/bin"
PHAR_URL="https://github.com/phpactor/phpactor/releases/latest/download/phpactor.phar"
TMP_PHAR="$(mktemp)"
if ! command -v "$PHP_BIN" >/dev/null 2>&1; then
    echo "php not found in PATH" >&2
    exit 1
fi
mkdir -p "${INSTALL_DIR}"
curl -fL "${PHAR_URL}" -o "${TMP_PHAR}"
chmod +x "${TMP_PHAR}"
mv "${TMP_PHAR}" "${INSTALL_DIR}/phpactor"
phpactor status || {
    echo "phpactor status failed; ensure ${INSTALL_DIR} is in PATH and PHP deps are OK." >&2
}
laravel_ls
Laravel LSP Reference
go install github.com/laravel-ls/laravel-ls/cmd/laravel-ls@latest

psalm_ls
Psalm LSP Reference
composer global require vimeo/psalm
phan_ls
Phan LSP Reference

```

```

composer require phan/phan


</div>
<strong>PlantUML</strong>

plantuml_ls
PlantUML LSP Reference

go install github.com/ptdewey/plantuml-lsp@latest


</div>
<strong>Postgresql</strong>

postgres_ls
<p>
    <a href="https://github.com/phan/phan">Postgres LSP Reference</a>
</p>

mkdir -p ~/.local/bin && cd ~/.local/bin
curl -L \
    https://github.com/supabase-community/postgres-language-server/releases/latest/download/po
    -o postgres-language-server
chmod +x postgres-language-server

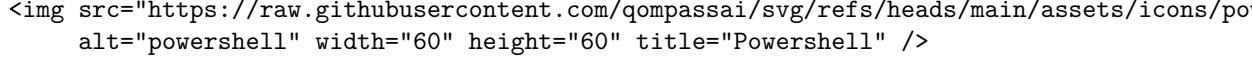
postrestoo_ls
PostgresTools LSP Reference

export REPO="supabase-community/postgres-language-server"
export INSTALL_DIR="$HOME/.local/bin"
export NAME="postgres-language-server"
arch=$(uname -m)
case "$arch" in
    x86_64|amd64) arch="x86_64" ;;
    aarch64|arm64) arch="aarch64" ;;
    *) echo "Unsupported arch: $arch" >&2; exit 1 ;;
esac
os=$(uname -s)
case "$os" in
    Linux) os="unknown-linux-gnu" ;;
    *) echo "This script is for Linux only."; exit 1 ;;
esac
bin="${NAME}_${arch}-${os}"
url="https://github.com/$REPO/releases/latest/download/$bin"
mkdir -p "$INSTALL_DIR"

```

```

tmp=$(mktemp)
trap 'rm -f "$tmp"' EXIT
if command -v curl >/dev/null 2>&1; then
    curl -L "$url" -o "$tmp"
else
    wget -O "$tmp" "$url"
fi
mv "$tmp" "$INSTALL_DIR/$NAME"
chmod +x "$INSTALL_DIR/$NAME"
echo "Installed to $INSTALL_DIR/$NAME"
echo "Ensure $INSTALL_DIR is in your PATH, then run:"
echo "  $NAME --help"



```

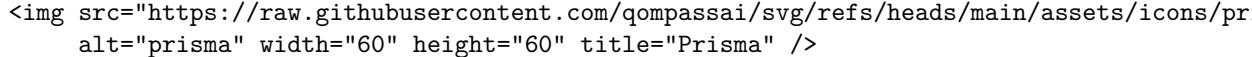
**Powershell**

pwershelles\_ls

Powershell LSP Reference

```

mkdir -p ~/.local/share/powershell-editor-services && \
cd ~/.local/share/powershell-editor-services && \
curl -s https://api.github.com/repos/PowerShell/PowerShellEditorServices/releases/latest | \
grep "browser_download_url.*PowerShellEditorServices.zip" | \
cut -d '"' -f 4 | \
xargs curl -L -o pses.zip && \
unzip -q pses.zip && \
rm pses.zip



```

**Prisma**

prisma\_ls

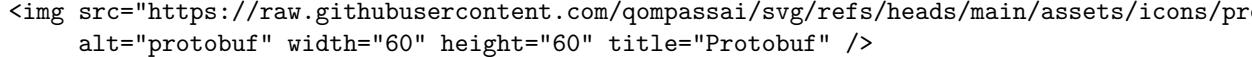
```

<p>
  <a href="https://www.npmjs.com/package/@prisma/language-server">Prisma LSP Reference</a>
</p>

```

pnpm add -g @prisma/language-server@latest

```



```

**Protobuf**

proto\_ls

## Protobuf LSP Reference

```
cargo install --git https://github.com/coder3101/protols
buf_ls

<p>
<a href="https://buf.build/docs/">Buf LSP Reference</a>
</p>

go install github.com/bufbuild/buf/cmd/buf@latest


</div>
<strong>Puppet</strong>

puppet_ls

<p>
<a href="https://github.com/puppetlabs/puppet-editor-services">Puppet LSP Reference</a>
</p>

export GEM_HOME="$HOME/.gem"
export GEM_PATH="$GEM_HOME"
export PATH="$HOME/.local/bin:$GEM_HOME/bin:$PATH"
git clone https://github.com/puppetlabs/puppet-editor-services.git && cd puppet-editor-servi
bundle exec rake gem_revendor


</div>
<strong>Python</strong>

basepy_ls

BasedPyright LSP Reference

pip install basedpyright

pyrefly_ls

<p>
<a href="https://pyrefly.org/">Pyrefly LSP Reference</a>
</p>

pip install pyrefly

ruff_ls

<p>
<a href="https://docs.astral.sh/ruff/">Ruff LSP Reference</a>
</p>

curl -LsSf https://astral.sh/ruff/install.sh | sh
```

ty\_ls

Ty LSP Reference

uv tool install ty

## OR

pip install ty

```

</div>
<strong>QT</strong>
```

qml\_ls

```
<p>
    <a href="https://doc-snapshots.qt.io/qt6-dev/qtqml-tooling-qmlls.html">QML LSP Reference</a>
</p>
```

```
curl -L \
    -o qmlls-workflow-0.5-linux-x86_64.tar.xz \
    https://github.com/TheQtCompanyRnD/qmlls-workflow/releases/download/0.5/qmlls-workflow-0.5
mkdir -p "$XDG_DATA_HOME/qmlls-workflow/0.5"
tar -xf qmlls-workflow-0.5-linux-x86_64.tar.xz \
    -C "$XDG_DATA_HOME/qmlls-workflow/0.5" \
ln -sf "$XDG_DATA_HOME/qmlls-workflow/0.5/bin/qmlls" \
    "$HOME/.local/bin/qmlls"
```

```

</div>
<strong>R</strong>
```

air\_ls

```
<p>
    <a href="https://posit-dev.github.io/air/integration-github-actions.html">Air LSP Reference</a>
</p>
```

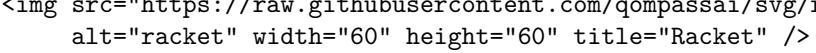
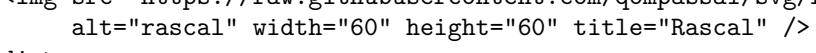
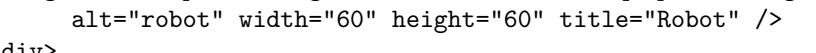
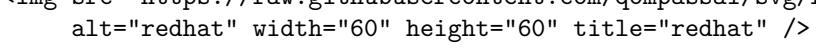
```
cargo install --git https://github.com/posit-dev/air air

</div>
<strong>Rescript</strong>
```

rescript\_ls

```
<p>
```

```

<a href="https://github.com/rescript-lang/rescript-vscode/tree/master/server">Rescript LSP Reference</a>
pnpm add -g @rescript/language-server@latest

</div>
<strong>Racket</strong>
racket_ls
<p>
  <a href="https://github.com/jeapostrophe/racket-langserver">Racket LSP Reference</a>
</p>
raco pkg install racket-langserver

</div>
<strong>Rascal</strong>
rascal_ls
<p>
  <a href="https://github.com/usethesource/rascal-language-servers">Rascal LSP Reference</a>
</p>
pnpm add -g @usethesource/rascal-vscode-dsl-lsp-server@latest

</div>
<strong>Robot</strong>
robotcode_ls
<p>
  <a href="https://github.com/dcermak/rpm-spec-language-server">Robotcode LSP Reference</a>
</p>
pip install robotcode[all]
robotframework_ls
<p>
  <a href="https://github.com/robocorp/robotframework-lsp">Robotframework LSP Reference</a>
</p>
pip install robotframework-lsp


```

```

</div>
<strong>Redhat Package Manager (RPM)</strong>
rpmspec_ls

<p>
  <a href="https://github.com/dcermak/rpm-spec-language-server">RPMSpec LSP Reference</a>
</p>

pip install rpm-spec-language-server


</div>
<strong>Ruby</strong>
rubocop_ls

<p>
  <a href="https://docs.rubocop.org/rubocop/1.81/index.html">Rubocop LSP Reference</a>
</p>

gem install rubocop

ruby_ls

<p>
  <a href="https://shopify.github.io/ruby-lsp/">Ruby LSP Reference</a>
</p>

gem install ruby-lsp

sorbet_ls

<p>
  <a href="https://sorbet.org/docs/lsp">Sorbet LSP Reference</a>
</p>

gem install sorbet sorbet-runtime

steep_ls

<p>
  <a href="https://github.com/soutaro/steep?tab=readme-ov-file">Steep LSP Reference</a>
</p>

gem install steep

standardrb_ls

<p>
  <a href="https://github.com/standardrb/standard">StandardRB LSP Reference</a>
</p>

gem install standard

```

```

stimulus_ls

    <p>
        <a href="https://github.com/marcoroth/stimulus-lsp">Stimulus LSP Reference</a>
    </p>

pnpm add -g stimulus-language-server@latest

stree_ls

    <p>
        <a href="https://ruby-syntax-tree.github.io/syntax_tree/">SyntaxTree LSP Reference</a>
    </p>

gem install syntax_tree

typeprof_ls

    <p>
        <a href="https://github.com/ruby/typeprof">TypeProf LSP Reference</a>
    </p>

gem install typeprof


</div>
<strong>Rust</strong>

bacon_ls

    <p>
        <a href="https://dystroy.org/bacon/">Bacon LSP Reference</a>
    </p>

cargo install --git https://github.com/Canop/bacon --features "clipboard sound"

    <p>
        <a href="https://github.com/pest-parser/pest-ide-tools">Pest Reference</a>
    </p>

cargo install --git https://github.com/pest-parser/pest-ide-tools

rustana_ls

    <p>
        <a href="https://scalameta.org/metals/docs/editors/user-configuration">Rust_Analyzer LSP R
    </p>

cargo install --git https://github.com/rust-lang/rust-analyzer rust-analyzer \
xtask proc-macro-srv-cli ungrammar2json


</div>

```

```

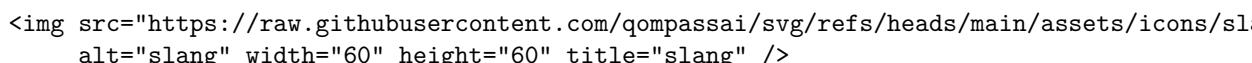
<strong>Scala</strong>
metals_ls
<p>
  <a href="https://scalameta.org/metals/docs/editors/user-configuration">Metals LSP Reference</a>
</p>
curl -fLo cs https://git.io/coursier-cli-"$(uname | tr LD ld)" && \
chmod +x cs && \
./cs install cs && \
rm cs && \
cs bootstrap \
  --java-opt -Xss4m \
  --java-opt -Xms100m \
  --java-opt -XX:+UseG1GC \
  --java-opt -XX:+UseStringDeduplication \
org.scalameta:metals_2.13:1.6.3 \
-o ~/bin/metals \
-f


</div>
<strong>Shell</strong>
bash_ls
<p>
  <a href="https://github.com/bash-lsp/bash-language-server">Bash LSP Reference</a>
</p>
pnpm add -g bash-language-server@latest
fish_ls
Fish LSP Reference
pnpm add -g fish-lsp@latest
nu_ls
<p>
  <a href="https://www.nushell.sh/">NuShell Reference</a>
</p>
cargo install --git https://github.com/nushell/nushell nu
#cargo install --git https://github.com/nushell/nushell nu_plugin_query \
#cargo install nu_plugin_formats --git https://github.com/nushell/nushell \
#cargo install nu_plugin_polars --git https://github.com/nushell/nushell
termux_ls

```

## Termux LSP Reference

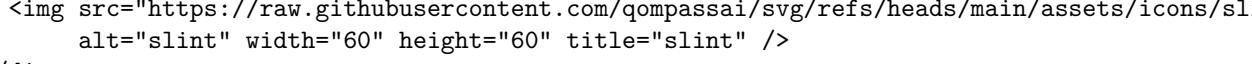
```
cargo install slint-lsp


</div>
<strong>Slang</strong>

slangd_ls

<p>
<a href="https://github.com/shader-slang/slang">Slang Reference</a>
</p>

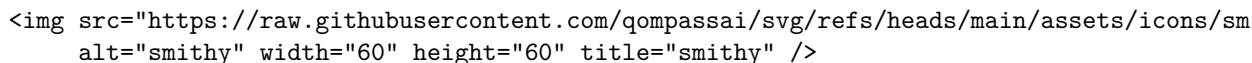
git clone https://github.com/shader-slang/slang.git --recursive && cd slang \
cmake -B build -S . \
-DCMAKE_BUILD_TYPE=Release \
-DSLANG_ENABLE_SLANGD=ON \
-DSLANG_ENABLE_SLANGC=ON \
-DSLANG_ENABLE_SLANGI=ON \
-DSLANG_ENABLE_SLANGRT=ON \
-DSLANG_ENABLE_GFX=ON \
-DSLANG_ENABLE_EXAMPLES=ON \
-DSLANG_ENABLE_TESTS=ON \
-DSLANG_ENABLE_SLANG_GLSLANG=ON \
-DSLANG_ENABLE_DXIL=ON \
-DSLANG_ENABLE_CUDA=ON \
-DSLANG_ENABLE_OPTIX=OFF \
-DSLANG_ENABLE_NVAPI=OFF \
-DSLANG_ENABLE_AFTERMATH=OFF \
-DSLANG_SLANG_LLVM_FLAVOR=FETCH_BINARY_IF_POSSIBLE && \
cmake --build build -j$(nproc) && sudo cmake --install build


</div>
<strong>Slint</strong>

slint_ls

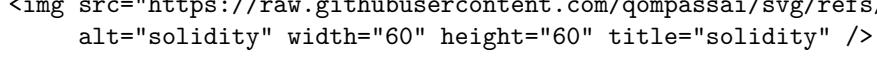
<p>
<a href="https://github.com/slnt-ui/slnt">Slint LSP Reference</a>
</p>

cargo install slint-lsp


</div>
<strong>Smithy</strong>
```

```

smithy_ls
Smithy LSP Reference

mkdir -p smithy-install/smithy && \
curl -L https://github.com/smithy-lang/smithy/releases/download/1.65.0/smithy-cli-linux- \
unzip -qo smithy-install/smithy-smithy-clilinux-x86_64.zip -d smithy-install && \
mv smithy-install/smithy-smithy-clilinux-x86_64/* smithy-install/smithy && \
curl -L 'https://github.com/smithy-lang/smithy/releases/download/1.65.0/smithy-clilinux-x86_64' -o \
sudo smithy-install/smithy/install && rm -rf smithy-install/ \


```

</div>

<strong>Solidity</strong>

solc\_ls

```

<p>
<a href="https://docs.soliditylang.org/en/latest/installing-solidity.html">Solc LSP Reference</a>
</p>

```

pnpm add -g solc@latest

solidity\_ls

```

<p>
<a href="https://github.com/qiuxiang/solidity-ls">Solidity LSP Reference</a>
</p>

```

pnpm add -g solidity-ls@latest

solang\_ls

Solang LSP Reference

cargo install solang

solidnomic\_ls

```

<img alt="SparQL icon" data-bbox="231 668 731 695"/>

```

</div>

<strong>SparQL</strong>

qlue\_ls

```

<p>
<a href="https://github.com/IoannisNezis/qlue-ls">Qlue LSP Reference</a>
</p>

```

cargo install --git https://github.com/IoannisNezis/qlue-ls

```

<img alt="SQL icon" data-bbox="231 828 681 855"/>

```

```

</div>
<strong>SQL</strong>
sq_ls

<p>
  <a href=" https://github.com/sqls-server/sqls">SQL LSP Reference</a>
</p>
go install github.com/sqls-server/sqls@latest


</div>
<strong>StarLark</strong>

starlark_ls

<p>
  <a href="https://github.com/facebookexperimental/starlark-rust/">Starlark LSP Reference</a
</p>
cargo install --git https://github.com/facebook/starlark-rust.git starlark_bin


</div>
<strong>Svelte</strong>

svelte_ls

Svelte LSP Reference

pnpm add -g svelte-language-server@latest


</div>
<strong>Systemd</strong>

systemd_ls

<p>
  <a href="https://github.com/JFryy/systemd-lsp">Systemd LSP Reference</a>
</p>
cargo install --git https://github.com/jfryy/systemd-lsp.git


</div>
<strong>Terraform</strong>

terraform_ls

```

```

<p>
  <a href="https://github.com/hashicorp/terraform-ls">Terraform LSP Reference</a>
</p>

go install github.com/hashicorp/terraform-ls@latest
#OR
git clone https://github.com/juliosueiras/terraform-lsp.git && cd terraform-lsp && nix-build
tflint_ls

<p>
  <a href="https://github.com/terraform-linters/tflint">TFLint LSP Reference</a>
</p>

curl -s https://raw.githubusercontent.com/terraform-linters/tflint/master/install_linux.sh | sh


</div>
<strong>TOML</strong>

taplo_ls

Taplo LSP Reference

cargo install --git https://github.com/tamasfe/taplo taplo-cli --features lsp
tombi_ls

Tombi LSP Reference

pip install tombi


</div>
<strong>TreeSitter-Query</strong>

tsquery_ls

TreeSitter-Query LSP Reference

cargo install --git https://github.com/ribru17/ts_query_ls ts_query_ls


</div>
<strong>Twig</strong>

twiggy_ls

Twiggy LSP Reference

pnpm add -g twiggy-language-server@latest

```

```


</div>
<strong>TypeScript</strong>
ts_ls
TypeScript LSP Reference
pnpm add -g typescript typescript-language-server@latest

</div>
<strong>TypeSpec</strong>
tsp_ls
TypeSpec LSP Reference
pnpm add -g @typespec/compiler@latest

</div>
<strong>Typst</strong>
tinymist_ls
TinyMist Typst LSP Reference
curl -sSL https://github.com/hongjr03/tinymist-nightly-installer/releases/latest/download/rust-tinymist-nightly-x86_64-unknown-linux-gnu.tar.gz | tar -xzf -

</div>
<strong>Vala</strong>
vala_ls
Vala LSP Reference
export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
export XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"
mkdir -p "$XDG_DATA_HOME/verible" "$XDG_BIN_HOME"
cd "$XDG_DATA_HOME/verible"
git clone https://github.com/chipsalliance/verible.git --recursive .
bazel build -c opt //...
.github/bin/simple-install.sh "$XDG_BIN_HOME"

</div>
<strong>Verilog</strong>

```

verible\_ls

Verible LSP Reference

```
export XDG_DATA_HOME="${XDG_DATA_HOME:-$HOME/.local/share}"
export XDG_BIN_HOME="${XDG_BIN_HOME:-$HOME/.local/bin}"
mkdir -p "$XDG_DATA_HOME/verible" "$XDG_BIN_HOME"
cd "$XDG_DATA_HOME/verible"
if [ ! -d .git ]; then
    git clone https://github.com/chipsalliance/verible.git --recursive .
else
    git pull --rebase
    git submodule update --init --recursive
fi
bazel build -c opt //...
bazel build -c opt :install-binaries
.github/bin/simple-install.sh "$XDG_BIN_HOME"
```

veridian\_ls

Veridian LSP Reference

```
cargo install --git https://github.com/vivekmalneedi/veridian.git --all-features
veryl_ls
```

Veryl LSP Reference

```
cargo install --git https://github.com/veryl-lang/veryl veryl-ls
svlang_ls
```

SVLang LSP Reference

```
pnpm add -g @imc-trading/svlangserver@latest

</div>
<strong>Vim</strong>
```

vim\_ls

Vim LSP Reference

```
pnpm add -g vim-language-server@latest

</div>
<strong>Vue</strong>
```

vue\_ls

Wasm Language Tools LSP Reference

```

pnpm add -g @vue/language-server@latest


</div>
<strong>WebAssembly (WASM)</strong>

wasmlangtoo_ls

Wasm Language Tools LSP Reference

cargo install --git https://github.com/g-plane/wasm-language-tools


</div>
<strong>WebGPU</strong>

wgslana_ls

WGSL Analyzer LSP Reference

cargo install --git https://github.com/wgsl-analyzer/wgsl-analyzer wgsl-analyzer


</div>
<strong>XML</strong>

lemminx_ls

Lemminx LSP Reference

git clone https://github.com/eclipse-lemminx/lemminx.git --recursive \
cd lemminx && ./mvnw clean verify


</div>
<strong>YAML</strong>

yaml_ls

Yaml LSP Reference

pnpm add -g yaml-language-server@latest

yamllint_ls

Yamllint LSP Reference

pip install yamllint

#OR

uv tool intsall yamllint

```

```

</div>
<strong>Zig</strong>
ziggy_ls
Ziggy LSP Reference
git clone https://github.com/kristoff-it/ziggy.git && cd ziggy && zig build -Doptimize=Release
ziggy_schema_ls
Ziggy LSP Reference
git clone https://github.com/kristoff-it/ziggy.git && cd ziggy && zig build -Doptimize=Release
z_ls
Zig LSP Reference
git clone https://github.com/zigtools/zls && cd zls && zig build -Doptimize=ReleaseSafe
```