Chapter 1

What you get

This project defines a language for the LaTeX 'listings' package that allows you to include Isabelle source code in your document, with all the funny symbols that Isabelle uses, and have it look much like it does in Isabelle's jedit window. In particular, you should be able to cut-and-paste from Isabelle to your LaTeX document and have things almost perfectly show up. Here's an example input:

```
\begin{lstlisting}
lemma times_assoc:
fixes a::nat and b and c
shows "P ≠ Q"
and "P ≠ l"
and "l∩ m = m"
and "∀P. P≤ P"
and "S → T"
and "⟦U⟧ ⇒ V" and "ℝ ≠ ℕ"
sorry
end
\end{lstlisting}
```

and the corresponding result:

```
lemma times_assoc:
fixes a::nat and b and c
shows "P ≠ Q"
and "P ≠ l"
and "l∩ m = m"
and "∀P. P≤ P"
```

```
and "S → T"
and "⟦U⟧ ⇒ V" and "ℝ ≠ ℕ"
sorry
end
```

And here's a list of all the characters that this language definition knows about, roughly corresponding to the various sections presented in the 'Symbols' tab of the jEdit interface, although if a symbol appears in two tabs, it only appears once here, so that (for instance) the 'Z notation' section has only a few symbols.

Digits

0123456789¼½¾0123

Abbrevs

›
‹

ZNotation

♯♀

Arrows

←←⇠↞→→

⇢↠⇐⇐

⇇⇉⇒⇛↔↭⇎⇔↦↦

⟝⟞↩↪←→↼⇀⇌⇝

↓↓ 1↑↟↓↡↕⇳

↣↠↠↠↠↠↦↦

Control

I⇩⇧

Control Block

Document

Greek

αβγδεζηθικλμνξπρστυφχψωΓΔΘΛΞΠΣΥΦΨΩ

Icon

Letter

ABCDEFGHIJKLM
NOPQRSTUVWXYZ

abcdefghijklm
nopqrstuvwxyz

𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐
𝔑𝔒𝔓𝔔ℜ𝔖𝔗𝔘𝔙𝔚𝔛𝔜ℨ

𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪
𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷

𝔸𝔹ℂ𝔻𝔼𝔽𝔾ℍ𝕀𝕁𝕂𝕃𝕄
ℕ𝕆ℙℚℝ𝕊𝕋𝕌𝕍𝕎𝕏𝕐ℤ

Logic

⊥⊤∧∧∨∨∀∃∄○□◇

```
Operator

↾◇∩⋂∪⋃⊔⨆⊓⨅∖∝⊎⨄±∓×÷⋅⋆∙∘⊕⨁⊗⨂⊙⨀⊖⊘∑∏∐∫∮⨿℧⤚⋙↿\⁀

Punctuation

∷⟨⟩⟪⟫⌈⌉⌊⌋⦇⦈⟦⟧⦃⦄◀▶»▯…⋯–⋅;●´‹›

Relation

⊢⊨⊩⊫⊣√≤≥≪≫≲≳⪅⪆∈∉

⬭⊆⊇⊏⊐⊑⊒≠∼≐≃≈≍≅⌣≡

⌢≺≻≼≽‖∥⦀⫽⊲⊳⊴⊵◃▹△≜

≀◁⧏▷⧐⦂E

Unsorted

⋈⋈†‡∞♣◇♡♠ℵ∅∇∂♭♮∠

º¶¿£°◊℘′ı

¸˝∊¤↩✓×⌂❖

End
```

The language definition has some limitations, discussed below.

1.1 Credit

This version of isabelle-listings was produced by John Hughes, but about 90 percent of it comes from the work of Jens Christoph Buerger, whose efforts I appreciate a great deal. Much credit also goes to various helpful folks on tex.stackexchange.com who helped me understand just enough about the challenges of dealing with character codes to get to the point where all this works.

1.2 Using this language definition

As in the example above, you merely open a listing, paste in stuff from the Isabelle/jEdit window, and close the listing.

1.2.1 Not the way to do it!

For most uses of Isabelle, this is the **wrong** way to produce source-code listings. The **right** way is to insert some magic instructions into the source code, use Isabelle's own build system to produce a LaTeX file, and then include 'snippets' from that file into whatever document you're producing. This is documented in the Isabelle build system manual, although as of a few months ago the 'comment' style shown there is no longer in use, and there are a couple of typos – excess backslashes – that make things fail.

That system seems complex — it is! — but it has the advantage that as your source code changes, so does your document. It has the disadvantage that by default it won't want to produce output for theories with errors in them, or may (depending on option-settings) suppress things with 'sorry' in them. If you're trying (as I am) to guide beginning users in the use of Isabelle, and want to demonstrate something being done wrong (and its correction!), this is actually somewhat annoying. And if you're writing a paper comparing multiple proof systems, having the Isabelle build-system in the middle of your document may be an annoyance rather than an aid.

The other disadvantage is that the LaTeX generated by the Isabelle build system isn't particularly readable: many characters are replaced by a LaTeX macro like `\isasymW` for a "mathcal" letter W, etc. The meanings of these macros is defined in a style file, and that results in a beautifully-formatted result, but hard-to-read LaTeX input. Of course, the notion is that you shouldn't be messing with that input at all once Isabelle produces it, but that's not how the document-preparation cycle always progresses for me.

1.3 Limitations

This project has several limitations.

- This project was developed using Isabelle2024 in early 2025. Isabelle will no doubt gradually add more symbols, and this will become obsolete.
- There are some symbols that cannot be used because they will appear as odd blocks in the output rather than as the symbol you wanted.

The font I've chosen to use (Julia) seems very well provisioned. If you try to shift to a different font, you may find that some symbols are not available.

- For reasons I don't understand, the Euro symbol (€) causes problems.
- Using this language-definition requires using LuaLaTeX or XeTeX (although I haven't tested it there) rather than pdfLaTeX.
- Using this language definition requires changing the input-characters allowed in your document. Many if not most of the unusual characters used in the jedit window have character codes outside of the normal range traditionally allowed by LaTeX. Some are in the range range 1000-2000, which may not be supported by all editors.
- As a particular example, Overleaf's current editor cannot handle many of these characters, so this language definition won't work there.
- Many characters in Isabelle's own style-definition require shifting vertically to look good. Almost all the arrows, for instance, are shifted somewhat. This language-definition does none of that.
- Isabelle uses arrows with many different lengths; this font does a bad job distinguishing among those.
- This language-definition is unsupported. I need it for a particular project, but do not anticipate maintaining it in the future.
- The failure of the 'listing' package to properly handle unicode without the whole '`lst@DefEC`' thing appears to be a bug. If the author of 'listings' fixes that bug, this language definition can be considerably simplified. I may actually do so if that fix is made.