Black blib2to3 to $NEW_PARSER

Lib2to3 is deprecated. Therefore, with blib2to3 being built on top of said deprecated library, it's not going to evolve with new language changes. Due to this, black will need to adopt a new base parsing library.

Purpose of this document

This document is to describe the problem of `lib2to3` deprecation and the need black has to move to a parser that will move forward as python gets new syntax. Black has some capital to potentially fund this work or a corporation with headcount could fund this work.

Due to the large Black user base, this work will benefit a large percentage of the Python Community. This work also ensures the long term maintainability of black and will allow the volunteer maintainers to continue to enhance Black.

Potential Parsers

Here are some potential libraries we could use to base newer versions of black parsing on.

libcst

Repo: https://github.com/Instagram/LibCST
Pros:

- None Yet

Cons:

- Doesn't support 3.9 or 3.10 yet
  - Or at least that's what the README says. It's based on Parso though which does support 3.9.
  - https://github.com/Instagram/LibCST/issues/285 for 3.10 support
- Ast does not always match https://github.com/Instagram/LibCST/issues/341

Parso

Repo: https://github.com/davidhalter/parso
Pros:

- Supports Python 3.9 https://github.com/davidhalter/parso/issues/153

Cons:

- Doesn't support 3.10 https://github.com/davidhalter/parso/issues/138

Maintain blib2to3

We could maintain blib2to3 and keep it up to date ourselves with new language changes.

Pros:

- Less work to migrate

Cons:

- Continuous work to add new language changes to blib2to3

Open questions

- Is the lib2to3 parser framework powerful enough to support features from the new PEG parser? CPython docs say lib2to3 is deprecated because it can't support PEG syntax.
    - According to @isidentical: it depends on the complexity though it is possible (at least as far as my experimentations go). Both parenthesized context managers and soft-keywords can be implemented, though in case of ambiguity it might require a bit more annotations (something like cut in the PEP 617, to commit to a rule). It is kinda harder when you try to do it on a parser with error handling (parso) but for black it should be easier to implement those 'hacks'.

Roll our own

It would be relatively easy to adapt the PEG parser generator used by CPython to generate Python instead of C -- that's how it was initially developed after all. We'd need to do something to preserve whitespace, but at least we can do it so that it exactly suits our purposes.

Pros:

- We won't have to wait on LibCST or Parso to support 3.10

Cons:

- Yet another thing to maintain

Open questions:

- The official grammar contains actions written in C that we can't reuse, and it contains error handling rules that we don't need.

I started making a full python parser from pegen and adding the missing bit and pieces to the python generator. The PR is WIP but I have been making good progress on it https://github.com/we-like-parsers/pegen/pull/11. Currently it simply aims at mimicking the C parser by generating the exact same AST but it could be adapted. - from Matthieu Dartiailh

Refactor / New Parser Introduction Design

Let's discuss here how it could be possibly introduced to black with minimal disruption.