Legend

- (TBD) = To Be Done
- (α) = To be tested
- (β) = Should be ready
- (OK) = OK.

Existing modules

Base64. No change that I can think of.

BitSet. Adding `of_enum` (TBD)

Dllist. No change that I can think of.

Enum. Going further in making `Enum` a complete replacement for `Stream` and preparing for a future parser combinator library based on `Enum`.

- Several additional constructors for `Enum.t`
  - `from_while`, similar to `from` but without exceptions (α)
  - `from_loop`, unfolder-style constructor (α)
  - `seq` and `seq_hide` unfolder-style constructor without exceptions (α)
  - `range`, abbreviated as `( -- )`
- Several simple combinators, inspired from SDFlow
  - `repeat`, `cycle`, `singleton` (α)
  - others ? (TBD)
- Several stream filters, again inspired from SDFlow or `ExtList`
  - `switch`, `switchn`, demultiplexing an enumeration (α)
  - `take_while`, `drop_while`, as in `ExtList`
- Several utility functions
  - `close`, to close an enum (α)
  - `before_do`, to add a lazy behavior to be triggered at the next use of an enum (α)
- Extending countability
  - an exception `Infinite_enum`, which may be triggered by `count` when it is known that the enumeration is infinite hence that `count` will fail (OK)

  - either adding a function `discarded`, with the number of enumeration elements which have been consumed, or adding a simple manner of creating an enumeration which can count itself, perhaps a variant of `init` (TBD) – this will be useful when writing parser combinators

- The "hidden" (i.e. public but undocumented) functions of Stream, used by Camlp4's stream parser syntax.

- A submodule `ExceptionLess` to provide exception-less replacement for exception-full functions. (α)

ExtArray. Need to import the documentation from Inria's documentation. (TBD)

ExtHashtbl. Need to import the documentation from Inria's documentation. (TBD)

ExtLib. Need to import the new modules. (TBD)

ExtList. Minor changes

- Additional common functions `index_of`, `rindex_of`, `index_ofq`, `rindex_ofq` (α)

- A few functions have been renamed : `nth` becomes `at`, `split_nth` becomes `split_at` `takewhile` becomes `take_while`, `dropwhile` becomes `drop_while`. The old functions remain in place but have been documented as obsolete. (β)

- The order of arguments in `remove_*` has been uniformized, but I might backtrack on that for compatibility purposes. (β)

- A submodule `ExceptionLess` to provide exception-less replacement for exception-full functions. (α).

- Need to import the documentation from Inria's documentation. (TBD).

ExtString. Minor additions

- Utilities `trim`, `splice` (β), `filter_map` (α), to simplify work-in-progress on Unicode.

- Need to import the documentation from Inria's documentation. (TBD)

Global. No change that I can think of.

Install. Need to compile the new modules. (TBD)

IO. A number of additions

- A number of utilities to read directly onto enumerations/write directly from enumerations (α).

- A few functions to open files, perform operations on them and then close the file (α).

- May need to check what else is needed to completely get rid of the standard library's channels (TBD).

Option. Trivial additions

- A function `enum` to map an option onto a trivial enumeration. (α)

- A function `get_exn`, variant on `get` but with the ability to specify another exception. (α)
- A fuction `bind` to allow monadic operations on monads. (TBD)

OptParse. No change I can think of.

PMap. No change I can think of.

UChar. ? (Edgar Friendly is working on that one, for better integration with ropes)

UTF8. ? (Edgar Friendly is working on that one, for better integration with ropes)

New modules

For most modules `Foo`, I plan to provide a corresponding module `FooLabels`.

Numbers

Number. Common interfaces for number types. (α)

Bool, Int... For each type `bool`, `int`, `int32`, `int64`, `native_int`, `big_int`, `float`, `complex`, provide a corresponding module with the most common operations, implementing the interfaces of `Number`. (α)

ExtChar. A few utilities for characters (character ranges, `is_whitespace`, etc.) (α)

Data structures

LazyList. Well, lazy lists, with essentially the same set of operations as `Enum`, just functional. (β)

ExtStream. Import of most of SDFlow, plus conversion to/from enumerations (α)

Ref. Trivial common operations on references (pre-incrementation, post-incrementation, etc.) (α)

Vect. Ropes as a general-purpose container (in progress)

Rope. Ropes of Unicode text (in progress)

ExtQueue. Add compatibility with `Enum`. (TBD)

ExtStack. Add compatibility with `Enum`. (TBD)

ExtBigArray. Add compatibility with `Enum`. (TBD)

Lexing & parsing

ExtGenlex. Port of `GenLex` to enumerations and lazy lists. (α)

ParserCo. Parser combinators for enumerations and lazy lists. (port of Parsec) (TBD)

Utilities

ExtRandom. Addition of enumerations and lazy lists of random values. (α) In the future, addition of random UTF8 characters from unicode ranges. (TBD)

Result. Similar to Haskell's Either monad. (in progress)

Later...

Extending everything string-based to make it compatible with ropes. This will probably mean rewriting plenty of functions of `Pervasives` and putting them into `Std`, plenty of functions of `Sys` and `Filename` and putting them into new modules `ExtSys` and `ExtFilename`, and possibly reworking `Printf` and `Scanf`.