Global objectives

Objectives

- Strongly-typed language
- Host-defined types
- High performance (Advanced programming techniques are not an objective)
- Syntax
    - Standardized
    - Nice and usable
- Debugging
    - It's a need of the project
    - GDB-compatible would be an interesting feature
- Independent compilation

Nonobjectives

- Dynamic programming functions
- Advanced programming techniques

Targets by September

Possible objectives for September 2014 (Bachelor's Project)

- Grammar and syntactic/lexical analyzer
- Typing
- Independent Compilation

Future objectives (could be postponed):

- GDB implementation and debugging
    - It's not a necessary work for the BP
- Strict performance
    - Not even a thing for a BP

Specification

- **Compiler**/interpreter (three possibilities)
  - Intermediate language
    - Benefits:
      - Better perfomance
      - Closer to host-code
    - Drawbacks:
      - Two languages to develop
      - A lot of compiling problems
    - Bonus: JIT compilable
  - Compile to host-code
    - Not a cake (uneasy)
    - It could attach it too much to host compilation
  - Interpretation
    - Easier alternative (1 language)
    - Worst performance
    - Bonus: Could be JIT compiled
- **Types**
  - It should be as basic as possible
  - Mainly arrays and tables, preferable to be compatible with C-typing
  - Could be interesting to have the typing independent to the compilation unit, as to have them dynamic on the current function/scope.
- **Functions** and flow
  - Further investigation will be needed:
    - Actual idea is to have a list of functions scheduled (no recursivity or stack?).
-

Project

The objective of this project is to **define, design** and **implement** a scripting language aimed to a particular use: logical evaluation of a series of values and states of a game, and the actions and events that happen after this evaluation has taken place. As such, the language **does not need** a lot of characteristics that are commonly available on all scripting languages (dynamic programming or weak typing), and is oriented to be based on the current game code design (**structs** and low-level functions that access the game).

It is also an objective to achieve **high performance**, because of the language also being aimed to be used on frame-to-frame operations such as **post-processing decisions** (choosing shader combinations for each element on-screen) or **gameplay logic** (damages, hits, states, history…). It is because of this limitation that this project exists: **common scripting languages** have a lot of functionality that **reduce the performance** and makes them unusable for this sort of project.