Parser Output

4-2-2012

Opening Discussion

- Minute essay comments:
  - Adding extra features to projects as misallocation of time.
- Traversing the default parse.

Specifying Output

- You can override the default of P by using P ^^ f. The f is a function (or partial function) that takes the normal output of P.
- The output you get is f(p).
- Example uses:
  - floatingPointNumber ^^ (_.toDouble)
  - “true” ^^ (x=>true)
  - “(“~ident~”,”~ident~”)” ^^ { case “(“~i1~”,”~i2~”)” => (i1,i2) }

Ignoring Parts of the Parse

- In something like the last example shown, there are strings that are part of the parse that really don't impact the result.
- When you have this type of situation you can use ~> or <~ instead of just ~. The parse result will only include what the arrow points to.
  - “(“~>ident~”,”~ident<~”)” ^^ { case i1~”,”~i2 => (i1,i2) }

Our Code

- Let's work on putting this type of functionality in our formula code.
- We want to parse to a tree similar to what we produced with the recursive parser we wrote ourselves.
- With that we can make this alternate code functional.

Minute Essay

- Questions? Can you think of anyplace you might use this in your project?