Retext

Design of an extensible system for analysing and manipulating natural language

Titus Wormer

```
'use strict';

var TextOMConstructor = require('textom'),
    ParseLatin = require('parse-latin');

function fromAST(TextOM, ast) {
    var iterator = -1,
        children, node, data, attribute;

    node = new TextOM[ast.type]();

    if ('children' in ast) {
        iterator = -1;
        children = ast.children;

        while (children[++iterator]) {
            node.append(fromAST(TextOM, children[iterator]));
        }
    } else {
        node.fromString(ast.value);
    }

    /* istanbul ignore if: TODO, Untestable, will change soon. */
    if ('data' in ast) {
        data = ast.data;

        for (attribute in data) {
            if (data.hasOwnProperty(attribute)) {
                node.data[attribute] = data[attribute];
            }
        }
    }

    return node;
}

function useImmediately(rootNode, use) {
    return function (plugin) {
        var self = this,
            length = self.plugins.length;

        use.apply(self, arguments);

        if (length !== self.plugins.length) {
            plugin(rootNode, self);
        }

        return self;
    };
}

/**
 * Define `Retext`. Exported above, and used to instantiate a new
 * `Retext`.
 *
 * @param {Function?} parser - the parser to use. Defaults to parse-latin.
 * @public
 * @constructor
 */
function Retext(parser) {
    var self = this;

    if (!parser) {
        parser = new ParseLatin();
    }

    self.parser = parser;
    self.TextOM = parser.TextOM = new TextOMConstructor();
    self.TextOM.parser = parser;
    self.plugins = [];
}

/**
 * `Retext#use` takes a plugin--a humble function--and when the parse
 * method of the Retext instance is called, the plugin will be called
 * with the parsed tree, and the retext instance as arguments.
 *
 * Note that, during the parsing stage, when the `use` method is called
 * by a plugin, the nested plugin is immediately called, before continuing
 * on with its parent plugin.
 *
 * @param {Function} plugin - the plugin to call when parsing.
 * @param {Function?} plugin.attach - called only once with a Retext
 *                                    instance. If you're planning on
 *                                    modifying TextOM or a parser, do it
 *                                    in this method.
 * @return this
 * @public
 */
Retext.prototype.use = function (plugin) {
    if (typeof plugin !== 'function') {
        throw new TypeError('Illegal invocation: \'' + plugin +
            '\' is not a valid argument for \'Retext.prototype.use\'');
    }

    var self = this,
        plugins = self.plugins;

    if (plugins.indexOf(plugin) === -1) {
```