This is the Readme file for EOClassBuilder Version 0.11.

EOClassBuilder supports the developer in the creation of Custom Enterprise Objects. On default EO's are objects of the class EOGenericRecord. This class is very usefull to map entites to a class.
But you can't use this class, if you want to implement special custom behavior. You have to create your own class, which implements this behavior. In the EOModeller application, you have to change the associated class of an entity from EOGenericRecord to the Name of your custom EO class. That means that you have to write the .h and .m file for your class by your own. If you have many custom EO classes and if they are big classes (containing 20 or more instance variables), this can take a long time.

For that reason I wrote EOClassBuilder. EOClassBuilder reads an .eomodel file and searches for custom classes. For every custom class found, it generates the correct interface and implementation file. The method names of the set and get methods are generated in the way the EOF needs them, so the default implementation of :

```
BOOL)takeValuesFromDictionary:(NSDictionary *)dictionary;
- (NSDictionary *)valuesForKeys:(NSArray *)keys;
```

accesses them directly.This saves a lot of time.

The following example is from sybase Pubs database.

I changed the class of the Author entity from EOGenericRecord to AuthorClass.

```
/*
 * Class      : AuthorClass
 * File       : AuthorClass.h
 * Author     : Generated by EOClassbuilder V 0.1
 * Date       : 1994-07-30 18:10:55 +0200
*/


#import <foundation/NSObject.h>

@class NSArray, NSString;


@interface AuthorClass : NSObject
{
      id      au_id;
      id      au_lname;
      id      firstName;
      id      toTitleAuthors;
}
```

```objc
- (id)init;
- (void)dealloc;
- (void)setAu_id:(NSString *)anNSString;
- (NSString *)au_id;
- (void)setAu_lname:(NSString *)anNSString;
- (NSString *)au_lname;
- (void)setFirstName:(NSString *)anNSString;
- (NSString *)firstName;
- (void)setToTitleAuthors:(NSArray *)anNSArray;
- (NSArray *)toTitleAuthors;

@end


/*
 * Class      : AuthorClass
 * File       : AuthorClass.m
 * Author     : Generated by EOClassbuilder V 0.1
 * Date       : 1994-07-30 18:10:55 +0200
 */


#import "AuthorClass.h"

#import <foundation/NSArray.h>
#import <foundation/NSString.h>


@implementation AuthorClass

- (id)init
{
      [super init];
      return self;
}

- (void)dealloc
{
      if ( au_id )
            [au_id release];
      if ( au_lname )
            [au_lname release];
      if ( firstName )
            [firstName release];
      if ( toTitleAuthors )
            [toTitleAuthors release];
      [super dealloc];
}

- (void)setAu_id:(NSString *)anNSString
{
      id      oldAu_id = au_id;

      if ( anNSString )
            au_id = [anNSString retain];
      else
            au_id = nil;
```

```objc
        if ( oldAu_id )
                [oldAu_id release];
}


- (NSString *)au_id
{
        return au_id;
}


- (void)setAu_lname:(NSString *)anNSString
{
        id      oldAu_lname = au_lname;

        if ( anNSString )
                au_lname = [anNSString retain];
        else
                au_lname = nil;

        if ( oldAu_lname )
                [oldAu_lname release];
}


- (NSString *)au_lname
{
        return au_lname;
}


- (void)setFirstName:(NSString *)anNSString
{
        id      oldFirstName = firstName;

        if ( anNSString )
                firstName = [anNSString retain];
        else
                firstName = nil;

        if ( oldFirstName )
                [oldFirstName release];
}


- (NSString *)firstName
{
        return firstName;
}


- (void)setToTitleAuthors:(NSArray *)anNSArray
{
        id      oldToTitleAuthors = toTitleAuthors;

        if ( anNSArray )
                toTitleAuthors = [anNSArray retain];
        else
                toTitleAuthors = nil;
```

```
        if ( oldToTitleAuthors )
                [oldToTitleAuthors release];
}


- (NSArray *)toTitleAuthors
{
        return toTitleAuthors;
}

@end
```

Now you can add or delete methods as you want and add your custom behavior.

Notes:
EOClassBuilder needs NEXTSTEP 3.2 (Black or White HW) and EOF Beta, otherwise it will not run.

Notes on Version 0.1:
In this version custom EO classes are generated as subclasses of NSObject.

Notes on Version 0.11:
No new features, just some bugs fixed. On some systems EOClassbuilder crashed. This was because a variable was autoreleased twice. I don't know why it crashes only on some systems and on others not.

Features for next versions:
1) Just generate classes selected in the Custom Class Browser.

2) Drag generated .h and .m files to other applications,
    e.g. ProjectBuilder. (through dragging a selection out
    of the "Custom Class Browser")

3) Let the Custom Class be a subclass of Object or NSObject

4) Specify custom naming conventions for set and get Methods

5) Add some methods from EOGeneric Record, e.g.
    - (void)setObject:anObject forKey:(NSString *)key;
    - objectForKey:(NSString *)key;
    - (void)removeObjectForKey:(NSString *)key;

6) Change the implementation of the setMethods

Bugs and suggestions:
Please send bug reports and suggestions to **em@marcon.de**. I hope there will be many suggestions and not to much bug reports. In the future I will extend the functionality and make EOClassBuilder more configurable.
(if there is time)

Licensing:

EOClassBuilder is a copyrighted work, distributed free of charge
for personal, educational, and commercial use.   It may not
be sold for profit, nor incorporated in products sold for
profit without the express permission of the author.   However,
recognized NeXT User Groups may include it on diskettes, tapes,
CD-ROMs, or other media priced above time-and-materials costs
for fundraising purposes.   This software may be placed on
Internet archive sites or other public software archives
provided that no "download" or other "enhanced services"
charges are assessed for access.

Only a nonexclusive license to use is granted; all other rights
are retained by the copyright holder.

Please do not redistribute modified versions of this package.

Source code availablility:

Source code is not currently available.

About MARCON:

Marcon - Evstathios Marinos Consulting is a privately held company.
Founded in 1988 we providing consulting, development & project management
services to our customers. Since 1991 we are providing also services in the
NEXTSTEP area. Our customers are small companies as well as worldwide
operating companies, like SwissBank, SEL/Alcatel and others.

If you like further information about our company and services please contact

us:

MARCON - Evstathios Marinos Consulting
c/o Mr. E. Marinos
Gartenstr. 2
D - 76133 Karlsruhe (GERMANY)
Phone : + 49 721 37 71 78
Fax : +49 721 37 71 79
E-Mail : info@marcon.de