Subject: Re: Anonymous delegates
Posted by Zardos on Wed, 07 Nov 2007 22:21:41 GMT
View Forum Message <> Reply to Message

Hi,

I have seen your foreach macro. Because I think it does not blend very good into the c++ syntax I would like to share my UPP-foreach version:

```
#define loop(v) \
 int MK__s = v; for(int _lv_ = MK__s; _lv_ > 0; _lv_--)

#define loopi(n, v) \
 int MK__s = v; for(int n = 0; n < MK__s; n++)

#define foreach(e, arr) \
 int MK__s = (arr).GetCount(); for(int _lv_ = 0; _lv_ < MK__s; _lv_++) \
 if(bool _foreach_continue = true) \
 for(e = (arr)[_lv_]; _foreach_continue; _foreach_continue = false)

#define foreach_n(n, e, arr) \
 int MK__s = (arr).GetCount(); for(int _lv_ = (n); _lv_ < MK__s; _lv_++) \
 if(bool _foreach_continue = true) \
 for(e = (arr)[_lv_]; _foreach_continue; _foreach_continue = false)

#define foreach_rev(e, arr) \
 for(int _lv_ = (arr).GetCount() - 1; _lv_ >= 0; _lv_--) \
 if(bool _foreach_continue = true) \
 for(e = (arr)[_lv_]; _foreach_continue; _foreach_continue = false)
```

Examples:

```
// repeat N times:
loop(10) {
   printf("Hello World\n");
}

repeat N times with index:
loopi(i, 10) {
   printf("Hello World %d\n", i);
}

// access container elements
Vector<int> vec;
foreach(int e, vec) {
   printf("e = %d\n", e);
```

```
}

// by ref:
Vector<int> vec;
foreach(int &e, vec) {
  printf("e = %d\n", e);
}

// in reverse order:
Vector<String> vec;
foreach_rev(const String &e, vec) {
  printf("e = %s\n", e);
}

// e declared outside:

Vector<int> vec;
int e;
foreach(e, vec) {
  printf("e = %d\n", e);
}

// e is only visible inside the foreach_scope:
Vector<String> vec;
foreach(const String &e, vec)
  printf("e = %s\n", e);

foreach_rev(const String &e, vec) // e used again
  printf("e = %s\n", e);
```

The macro produces "optimal" code if compiled in Release mode (VC++ / MINGW). Produces larger code in debug mode than a handcoded loop.

- Ralf
