**Fastcode C++ expression cheatsheet;**

Usage:

**fastcode** #scan all files in current dir and replaces fc expressions to C++ insertions

**fastcode -e expr** #generate C++ insertion from expression specified in expr. NOTE: no need for $ sign in expr

**fastcode -u** #undo all changes made by fc

**fastcode -c p/c/s** #scan all files and change case of declared functions, globals, namespaces and classes (pascal/camel/snake)

**fastcode -a codestyle.conf** #scan all files and change codestyle in them to codestyle described in codestyle.conf (syntax below)

Keys:

**-d dir**

**--dir=dir** #change scanning directory to dir (default .)

**-t cpp,hpp**

**--type=cpp,cc** #change scanned file types (default cpp,hpp,tcc)

**--codestyle=codestyle.conf**  #change codestyle describing file (default codestyle.conf)

Commands:

**$cClassname[field1,field2,field3]** - append class declaration

**$cClassname[field1,field2,field3]** - append class with definition

Generates class with fields described in []

Field description syntax:

**(!/+)(~~/~/&/>)(CVSu)[csilLfdDv](\*&)name(=literal)(@[nm]attr)**

**(!/+)** - access type (! - private, not specified - protected, + - public)

**(~~/~/&/>)** - getter and setter qualifiers (~~ - no setter, ~ - internal(no set/get), & - get by reference, no setter, > - initialization in constructor)

**(CVSu)** - C++ qualifiers (C - const, V - volatile, S - static) or u - unsigned

**[csilLfdDv]** - type (char, short, int, long, long long, float, double, long double, void)

**(\*&)** - pointer or reference

**(=literal)** - default member value or default parameter in constructor (depends on > qualifier)

**(@[nm]attr)** - field attribute

**n** - dynamic memory field allocated with new and deleted with delete. If attr specified, field will be allocated as array of **attr** elements

**M** - dynamic memory field allocated with malloc() and deleted with free(). If attr specified, field will be allocated as array of **attr** bytes

**m** - dynamic memory field allocated with malloc() and deleted with free(). If attr specified, field will be allocated as array of **attr** elements