

# Доработки библиотеки hpx

Вырастает из темы: [Доработки hpx](#)

[p0233](#) - 2017-10-15 HP

[p0566](#) - 2018-05-06 Proposed wording HP and RCU

[p1121](#) - 2019-01-20 Proposed wording and interface for HP

[folly](#) - Реализация, приближенная к p0566. Использует части библиотеки folly(SingletonThreadLocal, SingletonManager, folly:Executor).

[libcds](#) - Другая реализация HP.

folly умеет использовать не только thread\_local, libcds использует только thread\_local storage.

В p0233 написано “Due to the performance advantages of using TLS, the library implementation should allow the programmer to choose implementation paths that benefit from TLS when suitable, and avoid TLS when incompatible with the use case.”

libcds:

```
cds::Initialize(); once
cds::gc::HP hpGC; once
cds::threading::Manager::attachThread(); every thread
cds::threading::Manager::detachThread(); every thread
```

[http://libcds.sourceforge.net/doc/cds-api/index.html#cds\\_how\\_to\\_use](http://libcds.sourceforge.net/doc/cds-api/index.html#cds_how_to_use)

Не умеет в разные HP\_domain, умеет только в собственный thread\_local tls, не особо гибкий. Хотя умеет в

```
set_memory_allocator(
    void* ( *alloc_func )( size_t size ),    ///< \p malloc() function
    void( *free_func )( void * p ) )
```

class HP - главный класс, Before use any HP-related class you must initialize \p %HP by constructing \p %cds::gc::HP object in beginning of your \p main().

class Guard - A guard is a hazard pointer. Additionally, the Guard class manages allocation and deallocation of the hazard pointer.

```
Guard::protect(atomics::atomic<T> const& toGuard)
Guard::protect(atomics::atomic<T> const& toGuard, Func f)
```

Приводит T\* к void\* и работает с этими указателями.

From:

<http://wiki.osll.ru/> - **Open Source & Linux Lab**

Permanent link:

<http://wiki.osll.ru/doku.php/projects:hpx:start?rev=1589470331>

Last update: **2020/05/14 18:32**

