Итератор (шаблон проектирования)

**Iterator** ([англ.](http://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Cursor*) — [поведенческий](http://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D1%8B_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) [шаблон проектирования](http://ru.wikipedia.org/wiki/%D0%A8%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), относится к паттернам поведения. Представляет собой объект, позволяющий получить последовательный доступ к элементам объекта-агрегата без использования описаний каждого из объектов, входящий в состав агрегации.

Например, такие элементы как [дерево](http://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)), связанный список, хэш-таблица и массив могут быть пролистаны (и модифицированы) с помощью паттерна (объекта) *Итератор*.

Перебор элементов выполняется объектом итератора, а не самой коллекцией. Это упрощает интерфейс и реализацию коллекции, а также способствует более логичному распределению обязанностей.

Особенностью полноценно реализованного итератора является то, что код, использующий итератор, может ничего не знать о типе итерируемого агрегата.

Конечно же, почти любой агрегат можно итерировать указателем void\*, но при этом:

* не ясно, что является значением «конец агрегата», для двусвязного списка это &ListHead, для массива это &array[size], для односвязного списка это NULL
* операция Next сильно зависит от типа агрегата.

Итераторы абстрагируют именно эти 2 проблемы, используя полиморфный Next (часто реализованный как operator++ в С++) и полиморфный aggregate.end(), возвращающий значение «конец агрегата».

Таким образом, появляется возможность работы с диапазонами итераторов, при отсутствии знания о типе итерируемого агрегата. Например:

Iterator itBegin = aggregate.begin();

Iterator itEnd = aggregate.end();

func(itBegin, itEnd);

И далее:

void func(Iterator itBegin, Iterator itEnd)

{

for( Iterator it = itBegin, it != itEnd; ++it )

{

}

}