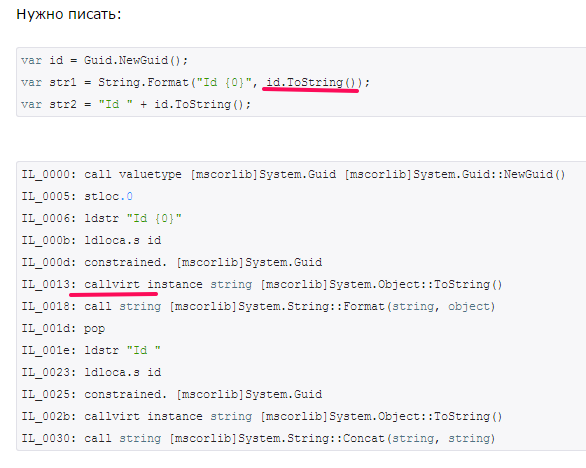
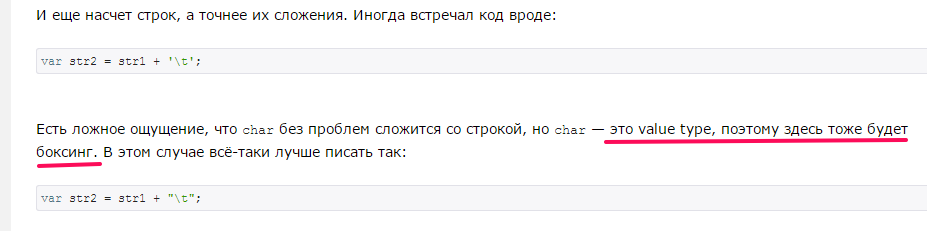
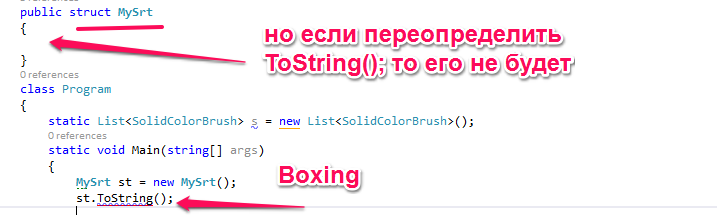
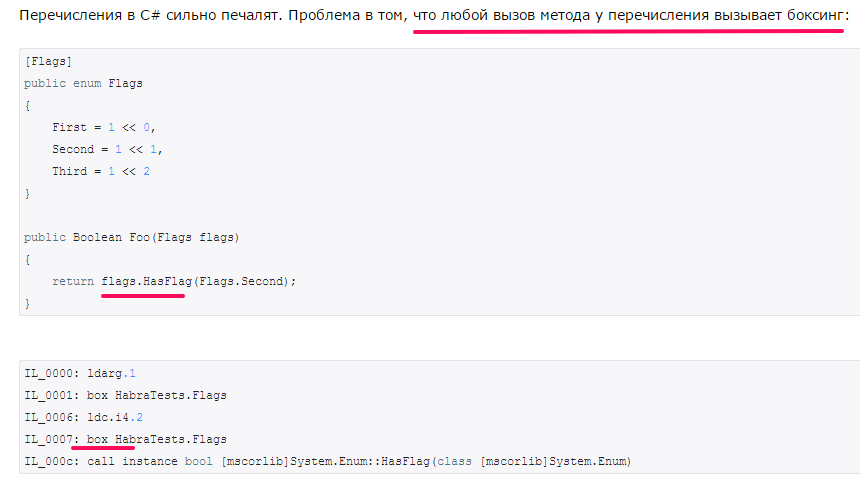
BOXING

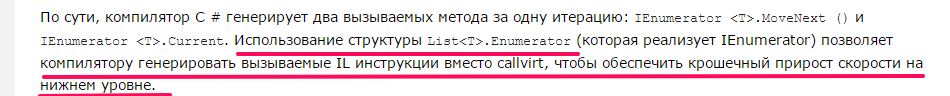








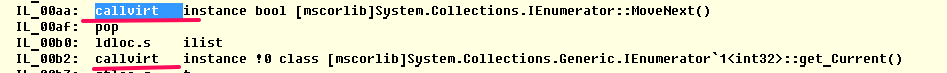


List<T>.**Enumerator**

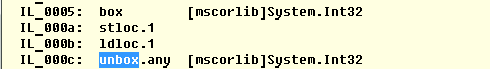
**В foreach приводить нельзя во избежание упаковки**

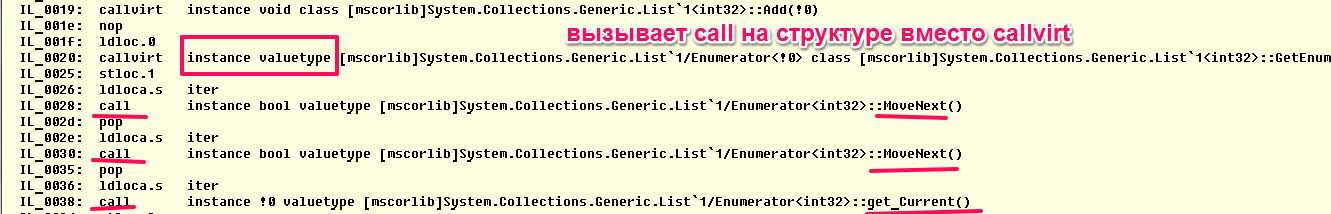


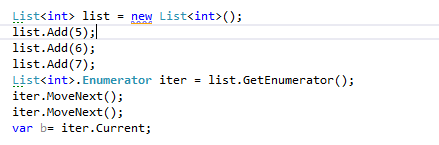
Еще и вызывает два callvirt



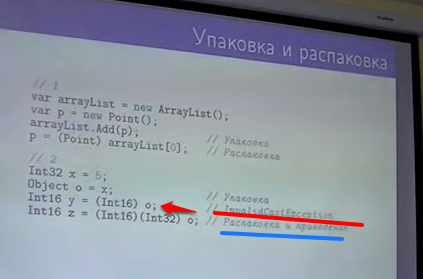
Отображение Box и Unbox



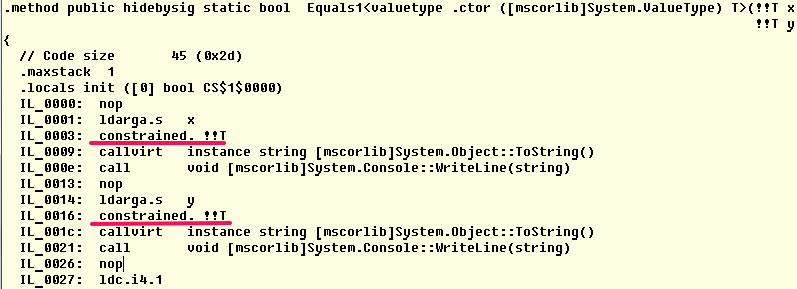




Не путать приведение типа к распаковку, синтаксис очень похож



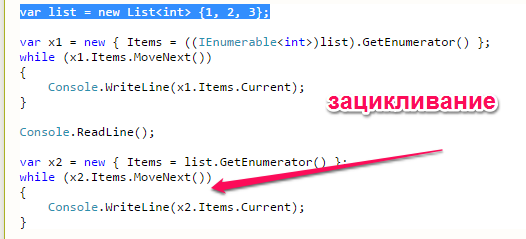
При generic методах можно указывать constrained



Происходит потому что вызывается свойство, а оно возвращает копию структуры

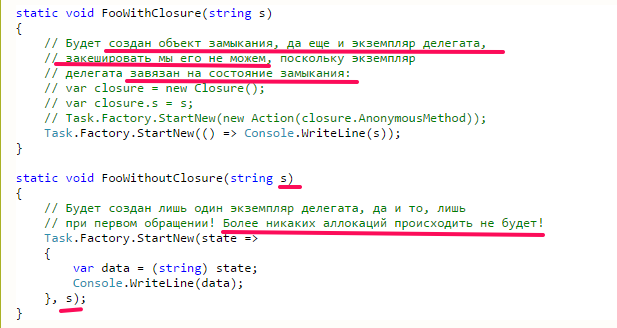
вызываем реализацию

List<T>.**Enumerator** GetEnumerator();

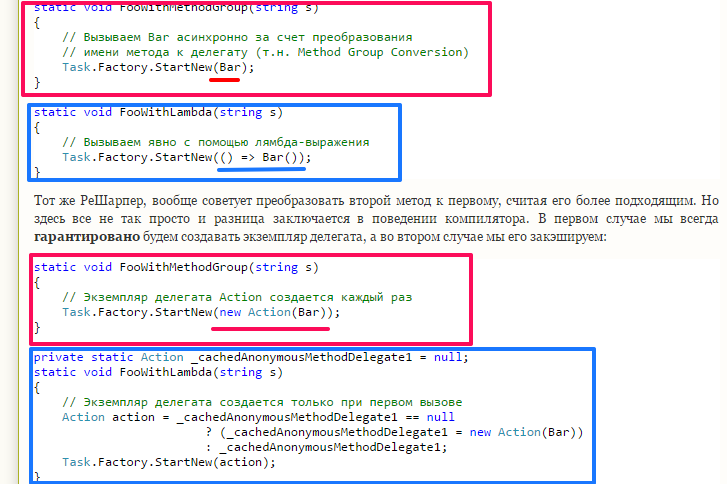


При передачи делегата нужно передавать лямду, чтобы ее можно было закешировать

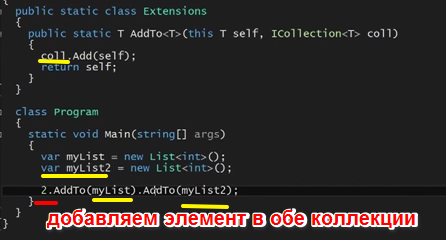
замыкание



обычное

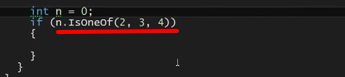


Инверсия вызовов



Проверяем находится ли элемент в коллеции





Или так проверка на наличие

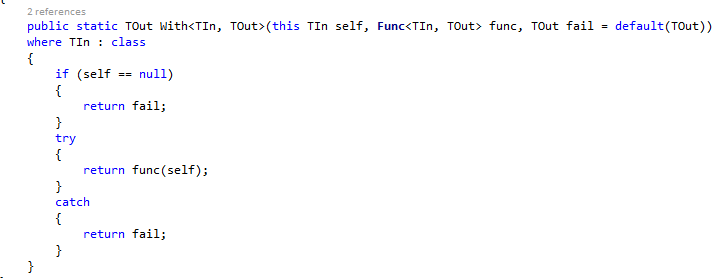


Без Проверки на null

Идентично p.Adress.City;



Заходит в первый раз класс Address (self (TIn)) который к примеру null **и** вызываем свойство City (становится TOut). Так как Address null возвращается Tout и на нем делаем то же самое



Подмена значений

