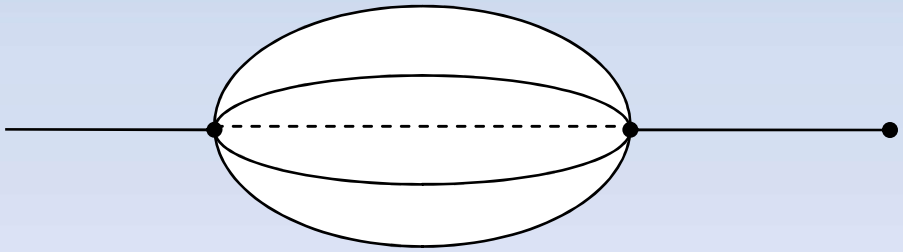
CountdownEvent

- Сигнальный объект синхронизации с внутренним счетчиком
- Применяется для блокировки одного потока в ожидании сигналов от других потоков
- При инициализации устанавливается начальное значение счетчика
- Каждый вызов метода **Signal** приводит к *потокобезопасному* уменьшению значения
- При достижении нуля ожидающий поток разблокируется

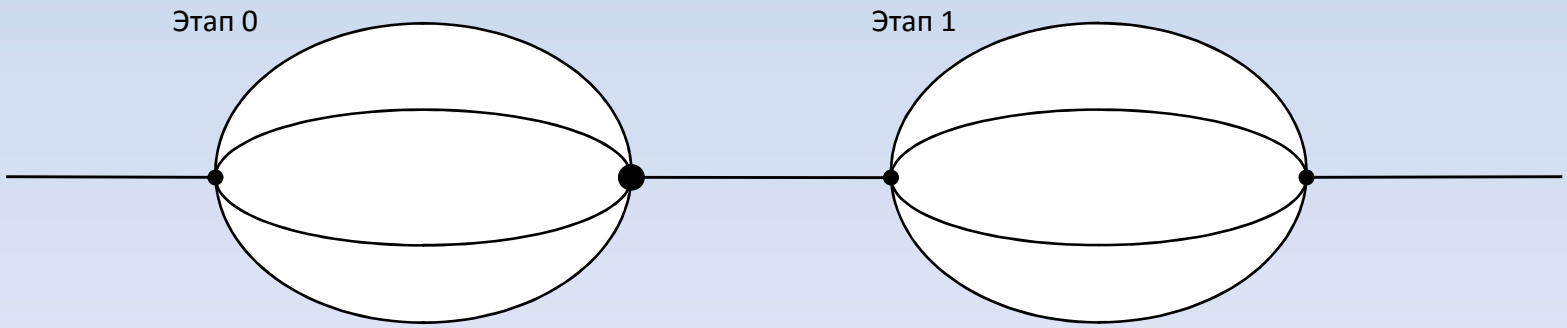


CountdownEvent

```
var counter = new CountdownEvent(3);
..
// Поточная функция
void f() {
    ..
    // сигнал = уменьшение счетчика
    counter.Signal();
}
void main() {
..
// ожидание обнуления счетчика
counter.Wait();
```

Барьерная синхронизация

- Барьер – точка синхронизации, при достижении которой потоки блокируются в ожидании остальных участников
- Многоэтапная барьерная синхронизация – обработка состоит из нескольких этапов; переход к следующему этапу осуществляется при завершении текущего этапа всеми участниками



Средство Barrier

- Инициализация барьера
  - Число участников;
  - Финальный обработчик после каждого этапа;
- Участники барьера
  - Вызов SignalAndWait определяет точку синхронизации: поток информирует о достижении барьера и ожидает подхода к барьеру всех остальных участников;
  - Могут получать текущую информацию:
    - Этап обработки - CurrentPhaseNumber
    - Число участников - ParticipantCount
    - Число оставшихся участников - ParticipantRemaining

```
// Инициализация
Barrier bar = new Barrier(3, (bar) =>
    Console.WriteLine(bar.CurrentPhaseNumber));

// работа потоков
void threadWork() {
    Work1();
    bar.SignalAndWait();
    Work2();
    bar.SignalAndWait();
    Work3();
  };
```

ReaderWriterLock

- Шаблон синхронизации «читатели-писатели»
- Потоки-читатели могут одновременно работать с разделяемым ресурсом
- Потоки-писатели требуют монопольного доступа к ресурса
- Дополнительно вводятся обновляющие потоки, осуществляющие преимущественно чтение, но в некоторых случаях требующие доступа для записи
- Объект ReaderWriterLock реализует шаблон «читателей-писателей» с помощью двух критических секций
- Существует облегченная версия с гибридной синхронизацией ReaderWriterLockSlim

```
// Инициализация объекта
var rw = new ReaderWriterLockSlim();
```

```
// читатели работают с ресурсом
rw.EnterReadLock();
        ..
rw.ExitReadLock();
```

```
// писатель монопольно
// работает с ресурсом
rw.EnterWriteLock();
..
rw.ExitWriteLock();
```

```
// читатели с возможностью
// записи
rw.EnterUpgradeableReadLock();
..
rw.EnterWriteLock();
..
rw.ExitWriteLock();
..
rw.ExitUpgradeableReadLock();
```