btSectorGrid의 첫 설계

중요한 전제 조건들

- b2Shape의 AABB는 섹터 하나에 들어갈 수 있다.
    - 인접한 9개의 섹터만 살펴보면 된다.
- CCD (Continuous Collision Detection)은 제공하지 않는다.
    - 필요시 Sweep 테스트 형태와 Shape 변경을 고려

인터페이스

b2SectorGrid

```
- QueryShape(const b2ShapeObject& obj, vector<void*> lst)
    - lst에 shape와 overlap 하는 userData를 넣어 돌려준다.
- QueryRay()

- Spawn(b2Shape, b2Filter, b2Transform, )
- Despawn()
- UpdateTransform(objectId, transform)
- UpdateTransform(objectId, position, rotation);
```

b2SectorObject
- b2Shape
- b2Filter
- b2Transform
- userData : void* (임의의 데이터 전달을 위해)
- 편의함수들
- <sector, proxy, bool> [4] 배열 유지

핵심 목표

- thread-safety
- performance
- efficient interface

Spawn/Despawn/Update 함수는 xlock이 필요하고, 놓인 섹터들에 대해서만
xlock을 사용해야 한다.

b2Shape의 사용 인터페이스

b2SectorGrid의 경우 변환 (회전과 이동) 반영이 잘 되어야 한다.

- AABB는 b2Transform에 따라 b2DynamicTree::MoveProxy()에서 반영
- 실제 contact 처리할 때 각 b2Shape에 변환이 적용되는 지 확인

b2CollideEdgeAndPolyon() 함수를 보면 두 개 대상의 변환을 합하여
각 b2Shape의 내부 정보인 정점과 중점을 옮겨서 테스트를 진행한다.

따라서, b2Contact::Evaluate()를 통해 처리하면 변환 값이 적용된
충돌 처리가 이루어진다.

이 내용은 세부 충돌 처리에도 있어야 하므로 양쪽에 적는다.

설계

b2SectorGrid

```
- b2Sector들을 std::unordered_map()으로 관리
    - Spawn() 시 해당 섹터가 없으면 동적으로 생성
    - nullSector를 내부에 갖고 Get 함수들에서 없으면 return
    - GetSectorByPosition(x, y)
    - GetSectorByIndex(index);

- b2SectorObject들 관리
    -

- Spawn()
    - b2SectorObject() 생성
    - GetSectorByPosition()
    - CreateSector()
    - b2Sector::CreateProxy()

- Despawn()
    - b2SectorObject()에서 현재 들어 있는 섹터들에 대해
        - b2Sector::DestroyProxy() 호출

- UpdateTransform(objId, b2Transform& tf)
    - 위치와 회전은 편의 함수
    - 새로운 AABB 계산
    - 인접한 9개의 Sector에 대해 겹치는 것 체크
    - 각 b2Sector에 대해 MoveProxy 호출
```

b2Sector

```
- b2DynamicTree를 내부에 가짐
- CreateProxy()
    - d2DynamicTree::CreateProxy() 호출
- MoveProxy()
    - d2DynamicTree::MoveProxy() 호출
- DestroyProxy()
    - d2DynamicTree::DestroyProxy() 호출
```

b2SectorGrid의 Query 함수

```
- Query의 Shape에 대해 AABB로 겹치는 섹터들에 대해
    - 각 섹터별로 AABB 검출로 겹치는 Object들 추출
    - 각 추출된 Object들에 대해 Contact 생성
        - 필요한 생성 함수들을 b2Contact::Create() 함수를 참고하여 추가
        - 각 Contact에 대해
            - Evaluate() 진행하여 세부 충돌 여부를 확인
            - b2Manifold의 Point() 카운트로 확인 가능
```

락 고려

```
- Spawn
    - 잠재적으로 b2SectorGrid에 대한 xlock 필요 (새로운 b2Sector 추가 시)
```

```
    - b2Sector에 대한 xlock 필요

- Despawn
    - b2SectorGrid에 대한 rlock
    - b2Sector에 대한 xlock

- UpdateTransform
    - b2SectorGrid에 대한 rlock
    - b2Sector에 대한 rlock
        - Sector 변경 시 xlock upgrade

- Query
    - b2SectorGrid에 대해 rlock
    - 연관된 b2Sector에 대해 rlock
```

락 특성

- 모든 락은 외부에 노출되지 않고 함수 내에서 처리하므로 데드락의 원인이 되지 않는다.
- UpdateTransform()에서 xlock이 꽤 자주 발생할 수 있다 (섹터 경계를 넘어 이동하는 경우)

테스트 설계

렌더링

test_bed의 DebugDraw에 lock을 추가한다.

- lines, triangles, points 각각에 대해 다른 락을 사용한다.
- Flush() 함수에서도 락을 사용한다.

렌더링 때문에 성능이 안 나올 수 있으므로 Disable 시키면 그리지 않도록 하고
통계 값을 테스트 종료 후 표시하도록 한다.

시뮬레이션

Entity 개체 수를 조절하면서 랜덤 워크를 하고, 일정 시간 범위에서
Query를 실행한다. Query를 실행할 때 해당 Shape과 결과를 그리도록 한다.

쓰레드 개수를 조절하여 쓰레드 개수에 따른 결과를 확인할 수 있게 한다.