WHAT'S NEXT GDC:06

Backwards Is Forward: Making Better Games with Test-Driven Development

Noel Llopis and Sean Houghton
High Moon Studios

GameDevelopers Conference

1. What is TDD?
2. How We Use TDD
3. TDD and Games
4. Lessons Learned
5. Wrap Up

GameDevelopers Conference

1. What is TDD?

(and why would I ever want to use it)

2. How We Use TDD

3. TDD and Games

4. Lessons Learned

5. Wrap Up

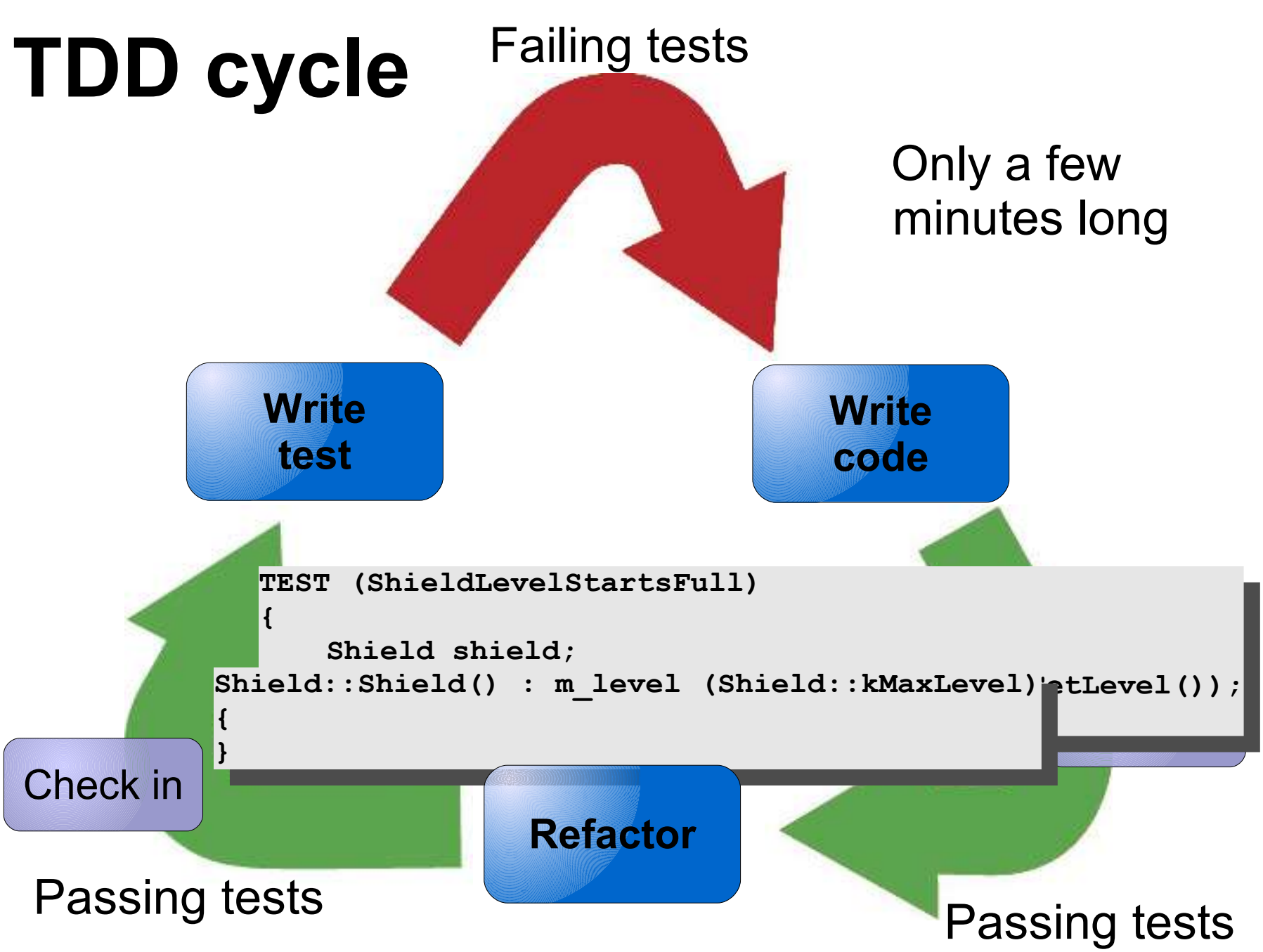
GameDevelopers Conference

```
define G(n) int n(int t, int q, int d)
#define X(p,t,s) (p>=t&&p<(t+s)&&(p-(t)&1023)<(s&1023))
#define U(m) *((signed char *)(m))
#define F if(!--q){
#define I(s) (int)main-(int)s
#define P(s,c,k) for(h=0; h>>14==0;
h+=129)Y(16*c+h/1024+Y(V+36))&128>>(h&7)?U(s+(h&15367))=k:k
G (B)
{
  Z;
  F D = E (Y (V), C = E (Y (V), Y (t + 4) + 3, 4, 0), 2, 0);
  Y (t + 12) = Y (t + 20) = i;
  Y (t + 24) = 1;
  Y (t + 28) = t;
  Y (t + 16) = 442890;
  Y (t + 28) = d = E (Y (V), s = D * 8 + 1664, 1, 0);
  for (p = 0; j < s; j++, p++)
    U (d + j) = i == D | j < p ? p--, 0 : (n = U (C + 512 + i++))
' ? p |=
      n * 56 - 497, 0 : n;
}
n = Y (Y (t + 4)) & 1;
F
U (Y (t + 28) + 1536) |=
62 & -n;
M
U (d + D) =
X (D, Y (t + 12) + 26628, 412162) ? X (D, Y (t + 12) + 27653,
                                       410112) ? 31 : 0 : U (d + D
for (; j < 12800; j += 8)
  P (d + 27653 + Y (t + 12) + ' ' * (j & ~511) + j % 512,
     U (Y (t + 28) + j / 8 + 64 * Y (t + 20)), 0);
}
F if (n)
  {
    D = Y (t + 28);
    if (d - 10)
      U (++Y (t + 24) + D + 1535) = d;
    else
      {
        for (i = D; i < D + 1600; i++)
          U (i) = U (i + 64);
        Y (t + 24) = 1;
        E (Y (V), i - 127, 3, 0);
      }
  }
else
```


DO NOT PROCEED
PAST THIS POINT
UNSTABLE CLIFFS

TDD addresses those problems

TDD cycle


Failing tests
Only a few minutes long
Write test
Write code
TEST (ShieldLevelStartsFull)
{
Shield shield;
Shield::Shield() : m_level (Shield::kMaxLevel)etLevel());
{
}
Check in
Refactor
Passing tests
Passing tests

Benefits: Simplicity, modularity

Benefits: Safety net

Benefits: Instant feedback

Milestone: ~2 months

Iteration: 2-4 weeks

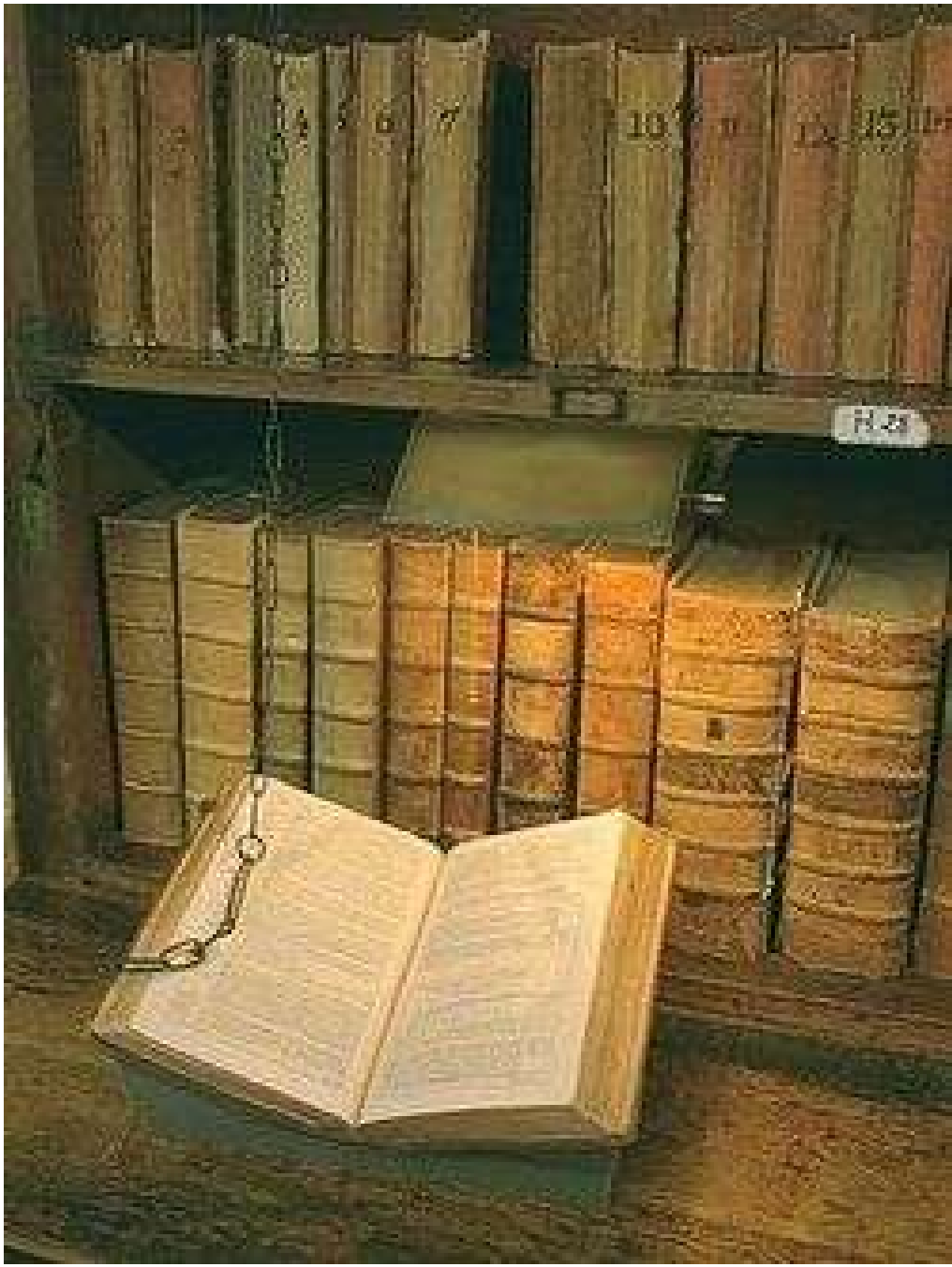
Nightly build: 1 day

Automated builds: ~1 hour

TDD: 30s to 3-4 minutes

```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

READY.
LOAD"*",8,1

SEARCHING FOR *
LOADING
READY.
RUN
```

Benefits: Documentation

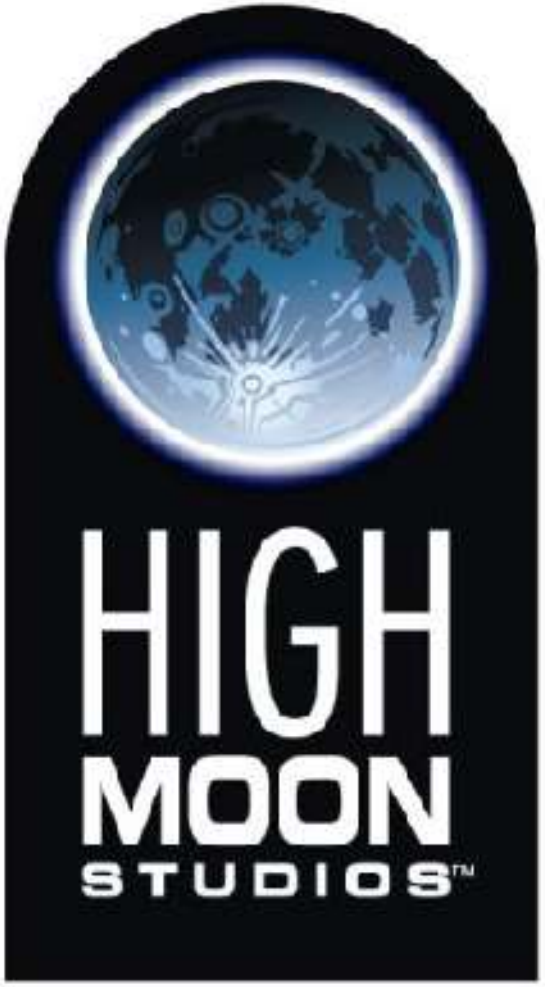


TDD != Unit tests

TDD != Testing strategy

TDD == Development technique

Use of TDD in games



More? Let us know.

~~1. What is TDD?~~

2. How We Use TDD

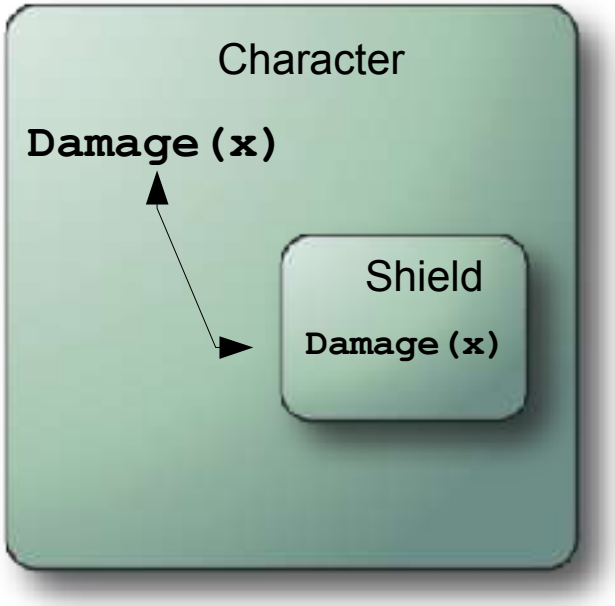
3. TDD and games

4. Lessons Learned

5. Wrap up

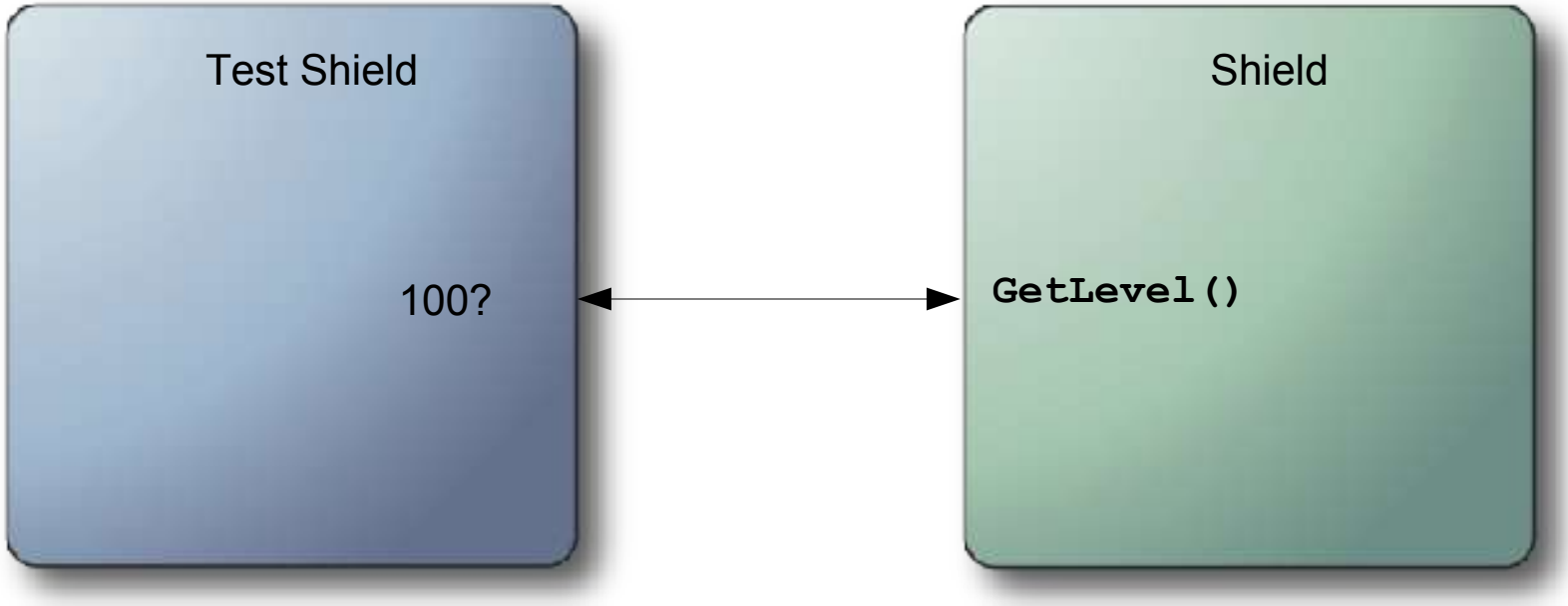
GameDevelopers Conference

Character + Shield




```
class Character
{
    IShield* m_shield;
public:
    Character();
    void Damage(float amount);
    float GetHealth() const;
};
```

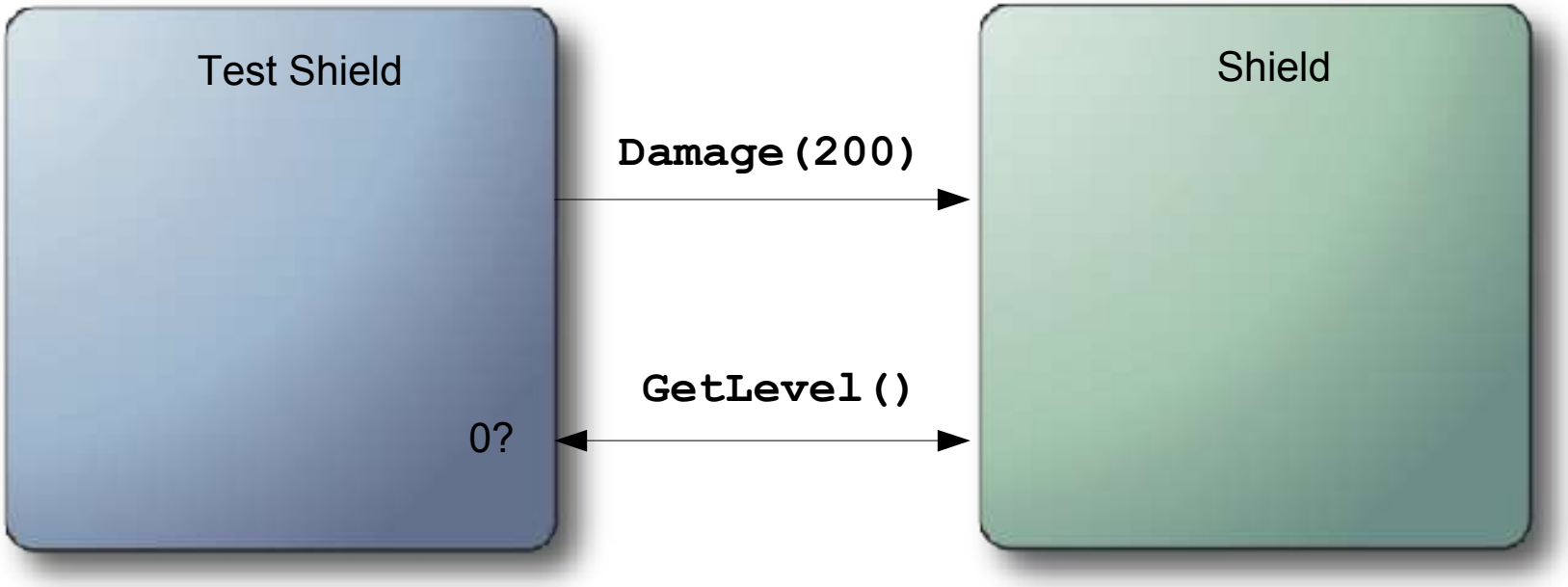
Testing Return Values




```
TEST (ShieldLevelStartsFull)
{
    Shield shield;
    CHECK_EQUAL (100, shield.GetLevel());
}
```

"Failure in ShieldLevelStartsFull: Expected 100 but got 0"

Testing State

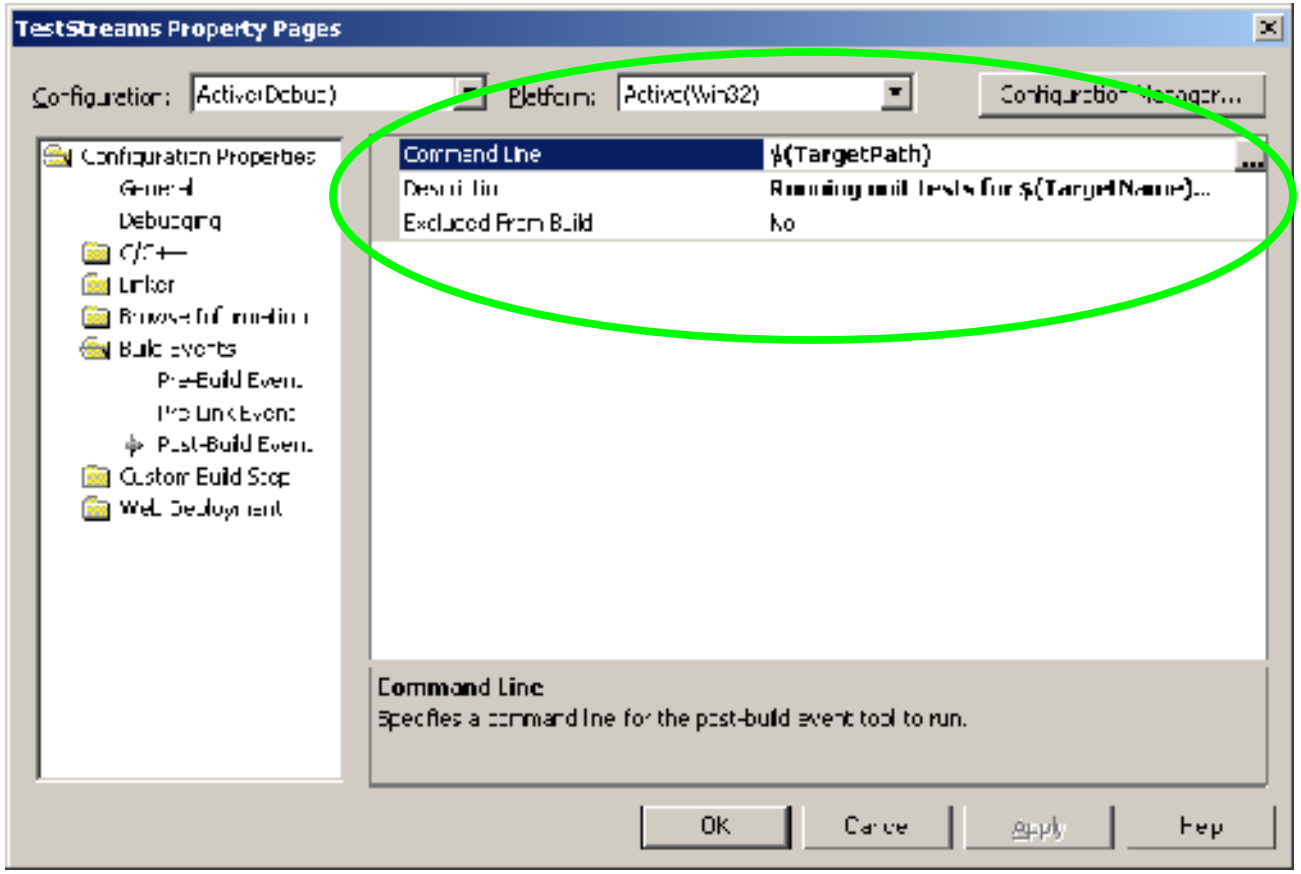



```
TEST (LevelCannotBeDamagedBelowZero)
{
    Shield shield;
    shield.Damage(200);
    CHECK_EQUAL (0, shield.GetLevel());
}
```

Where do we put tests?

- TestGame.exe (links with Game.lib)
- `#ifdef UNIT_TESTS`
- GameTests.DLL
- GameTests.upk

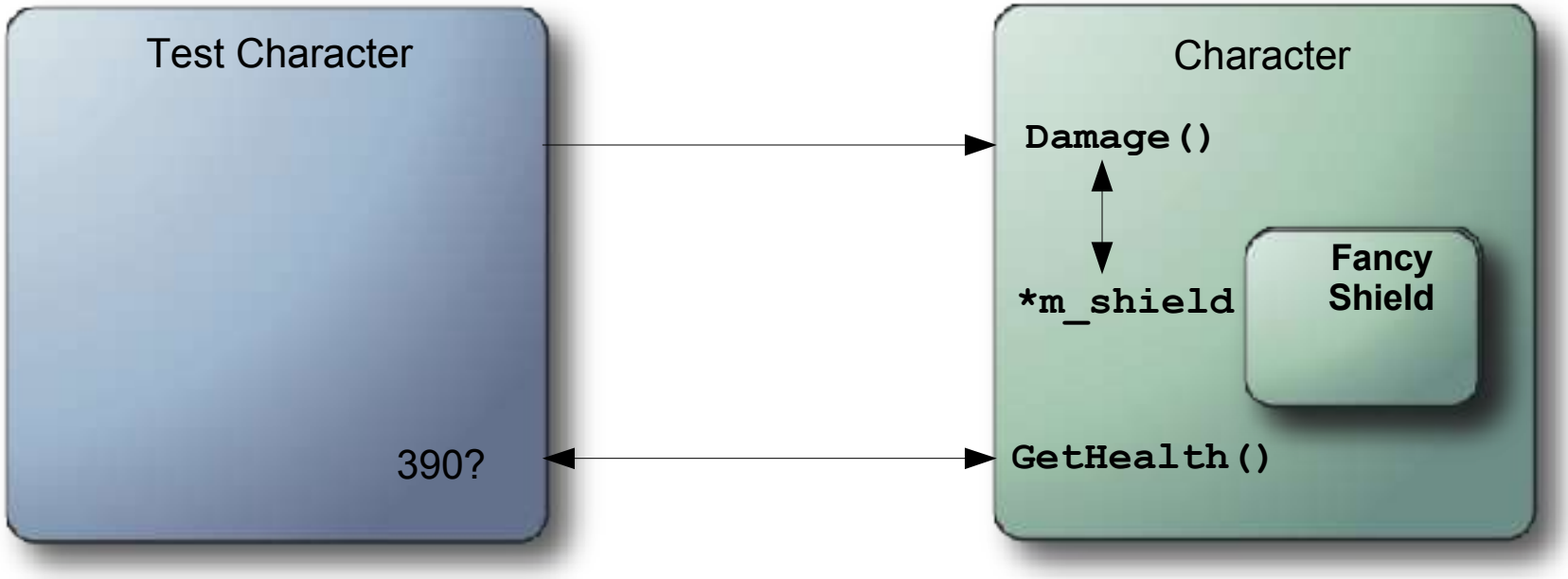
Run Tests With Every Build



TDD DEMO

Testing Interaction

(Can Be A Problem Initially)

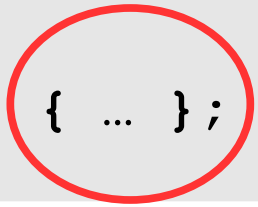



```
TEST(CharacterUsesShieldToAbsorbDamage)
{
    Character character(400);
    character.Damage(100);
    CHECK_EQUAL(390, character.GetHealth());
}
```

```
class IShield
{
public:
   float Damage(float amount) = 0;
}
```

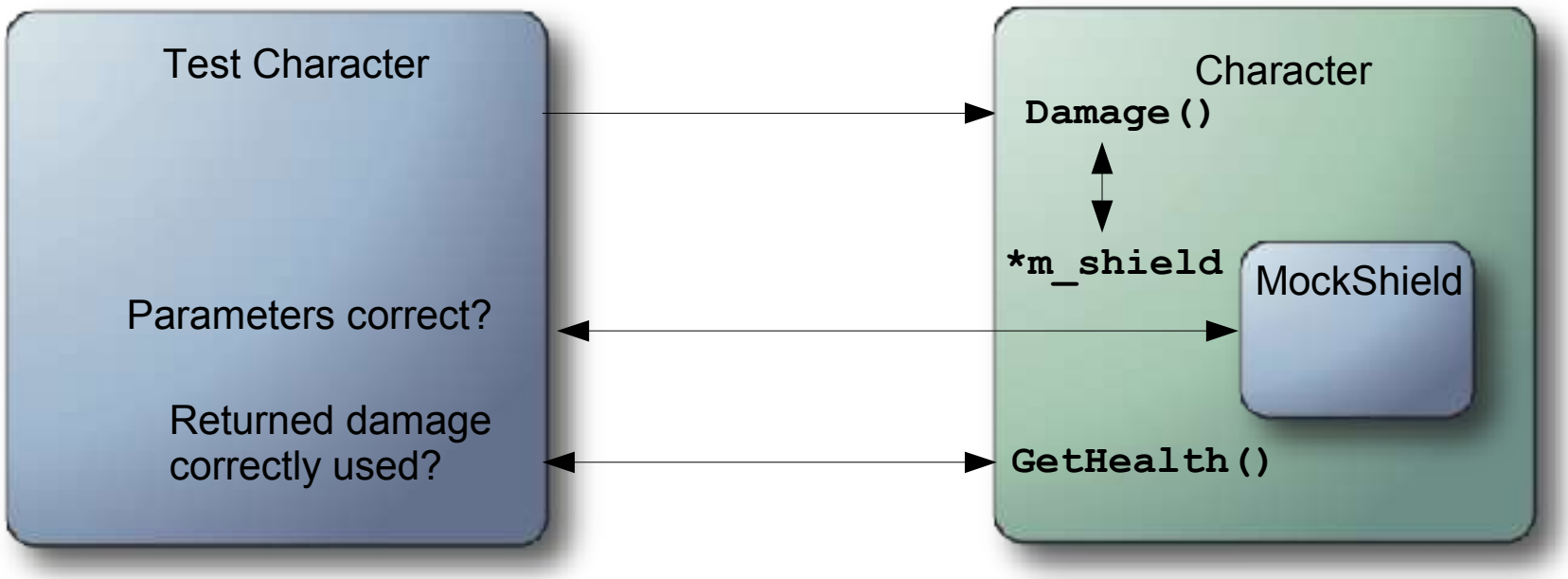
A mock object stands in for an object outside the unit you're testing

```
class FancyShield : public IShield
{
public:
   float Damage(float amount) { … };
}
```



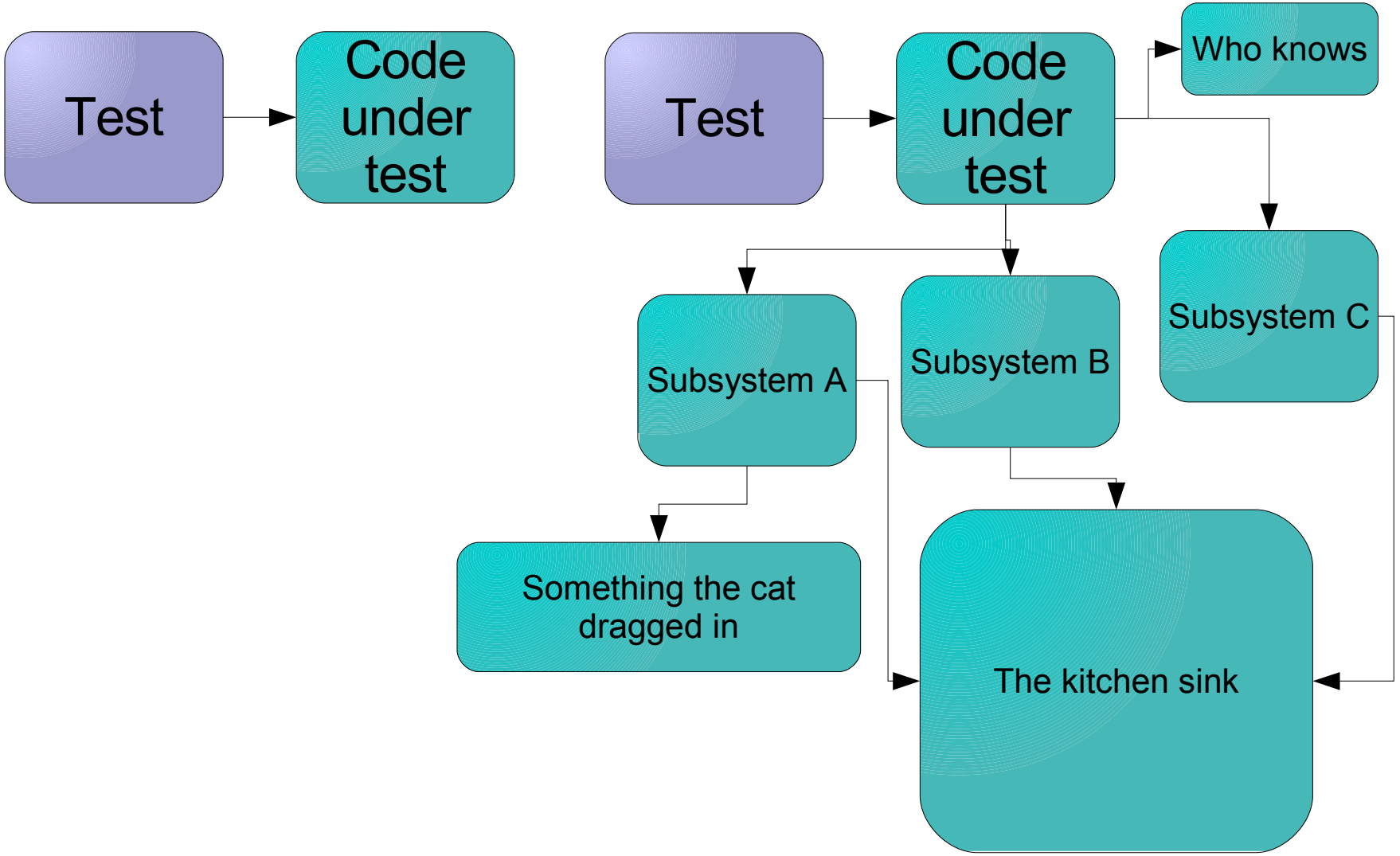
```
class MockShield : public IShield
{
public:
   float damagePassedIn;
   float damageToReturn;
   float Damage(float amount)
   {
      damagePassedIn = amount;
      return damageToReturn;
   }
}
```

Using a Mock in Your Test




```
TEST(CharacterUsesShieldToAbsorbDamage)
{
  MockShield mockShield = new MockShield;
  mockShield->damageToReturn = 10;
  Character character(400, mockShield);

  character.Damage(200);

  CHECK_EQUAL(200, mockShield->damagePassedIn);
  CHECK_EQUAL(390, character.GetHealth());
}
```

Best Practices: Test only the code at hand



```
TEST(ActorDoesntMoveIfPelvisBodyIsInSamePositionAsPelvisAnim)
{
    component = ConstructObject<UAmpPhysicallyDrivableSkeletalComponent>();
    component->physicalPelvisHandle = NULL;
    component->SetOwner(owner);
    component->SkeletalMesh = skelMesh;
    component->Animations = CreateReadable2BoneAnimSequenceForAmpRagdollGetup(component, skelMesh, 10.0f, 0.
    component->PhysicsAsset = physicsAsset;
    component->SpaceBases.AddZeroed(2);
    component->InitComponentRBPhys(false);
    component->LocalToWorld = FMatrix::Identity;
    const FVector actorPos(100,200,300);
    const FVector pelvisBodyPositionWS(100,200,380);
    const FTranslationMatrix actorToWorld(actorPos);
    owner->Location = actorPos;
    component->ConditionalUpdateTransform(actorToWorld);
    INT pelvisIndex = physicsAsset->CreateNewBody(TEXT("Bone1"));
    URB_BodySetup* pelvisSetup = physicsAsset->BodySetup(pelvisIndex);
    FPhysAssetCreateParams params = GetGenericCreateParamsForAmpRagdollGetup();
    physicsAsset->CreateCollisionFromBone(pelvisSetup,
                                skelMesh,
                                1,
                                params,
                                boneThings);

    URB_BodyInstance* pelvisBody = component->PhysicsAssetInstance->Bodies(0);
    NxActor* pelvisNxActor = pelvisBody->GetNxActor();
    SetRigidBodyPositionWSForAmpRagdollGetup(*pelvisNxActor, pelvisBodyPositionWS);

    component->UpdateSkelPose(0.016f);
    component->RetransformActorToMatchCurrrentRoot(TransformManipulator());

    const float kTolerance(0.002f);

    FMatrix expectedActorMatrix;
    expectedActorMatrix.SetIdentity();
    expectedActorMatrix.M[3][0] = actorPos.X;
    expectedActorMatrix.M[3][1] = actorPos.Y;
    expectedActorMatrix.M[3][2] = actorPos.Z;
    const FMatrix actorMatrix = owner->LocalToWorld();
    CHECK_ARRAY2D_CLOSE(expectedActorMatrix.M, actorMatrix.M, 4, 4, kTolerance);
}
```

Best Practices: Keep Tests Fast

```
Slow Test(24 > 20 ms): CheckSpotOverlapIsHandledCorrectly1Test
Slow Test(25 > 20 ms): CheckSpotOverlapIsHandledCorrectly2Test
Slow Test(24 > 20 ms): DeleteWaveEventFailsIfEventDoesntExistInCueTest
Slow Test(22 > 20 ms): CanGetObjectsInBrowserListPackageTest
Slow Test(48 > 20 ms): HmAddActorCallsCreateActorTest
Slow Test(74 > 20 ms): HmReplaceActorDoesNothingIfEmptySelectionTest
Slow Test(57 > 20 ms): HmReplaceActorWorksIfTwoActorsSelectedTest
Slow Test(26 > 20 ms): ThrowExceptionWhenTrackIndexOutOfRangeTest

Total time spent in 1923 tests: 4.83 seconds.

Time spent in 26 slow tests: 2.54 seconds.
```

Best Practices: Keep Tests Fast

```
Running unit tests TestDebugServer in Debug...
116 tests run
There were no test failures. Test time: 0.016 seconds.

Running unit tests for TestStreams in Debug...
138 tests run
There were no test failures. Test time: 0.015 seconds.

Running unit tests TestMath in Debug...
245 tests run
There were no test failures. Test time: 0.001 seconds.

Running unit tests...
184 tests run
There were no test failures. Test time: 0.359 seconds.
```

Best Practices: Keep Tests Independent

```
g_CollisionWorldSingleton
```

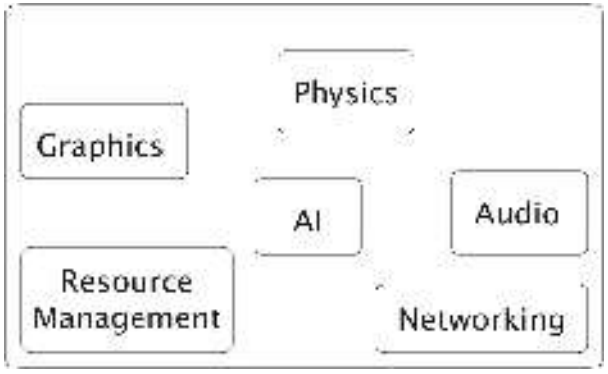
~~1. What is TDD?~~

~~2. How We Use TDD~~

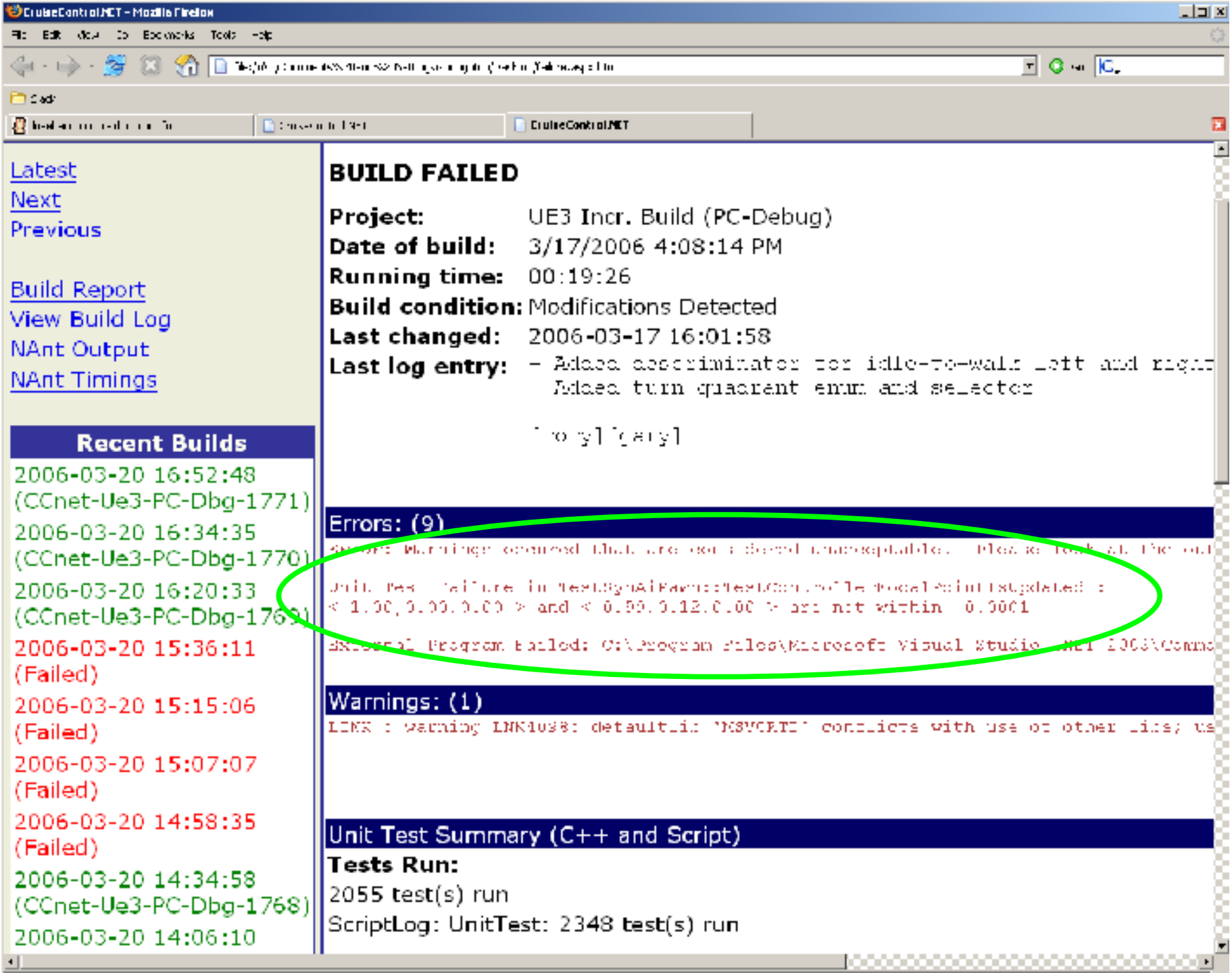
3. TDD and Games

4. Lessons Learned

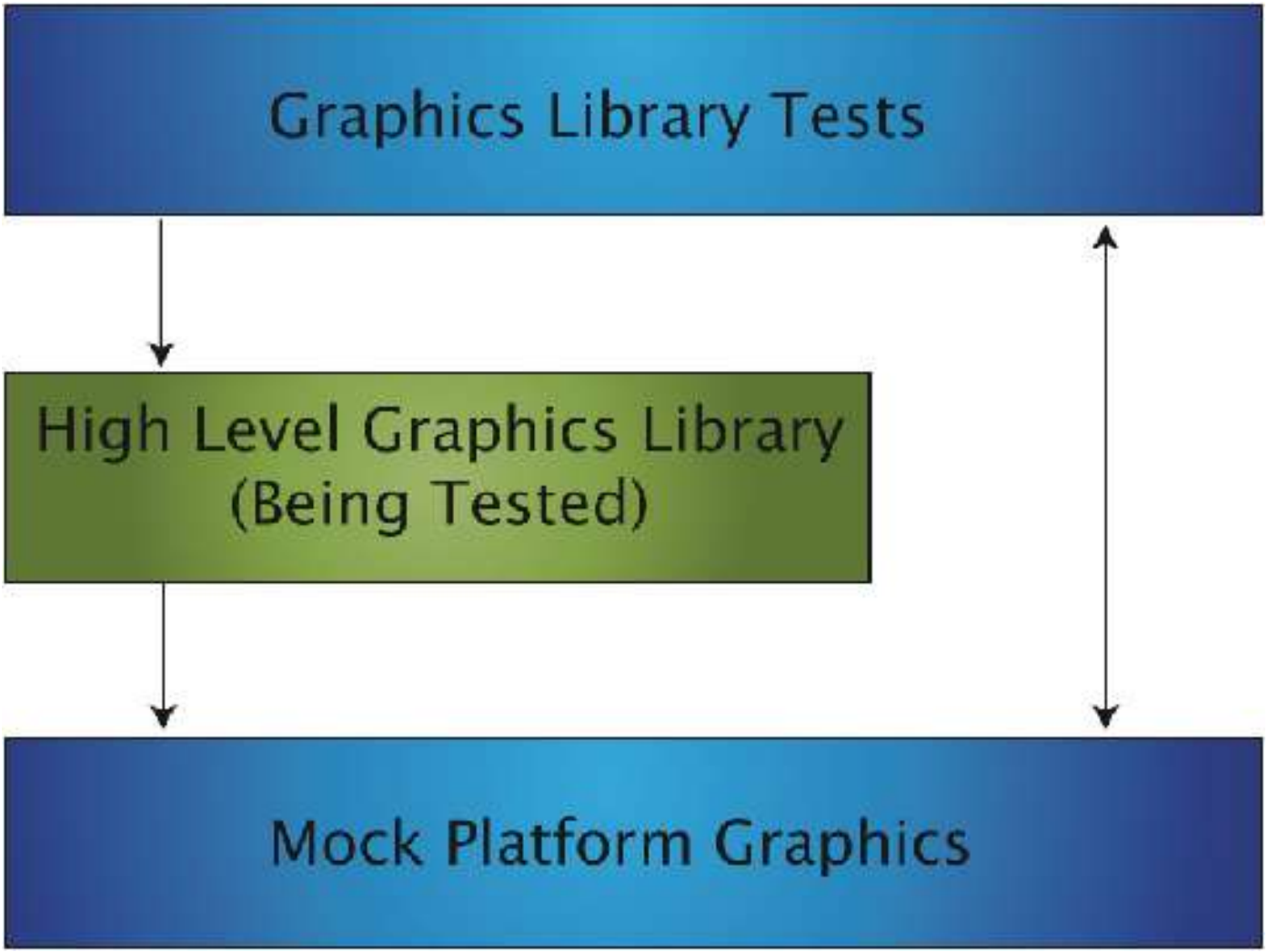
5. Wrap Up



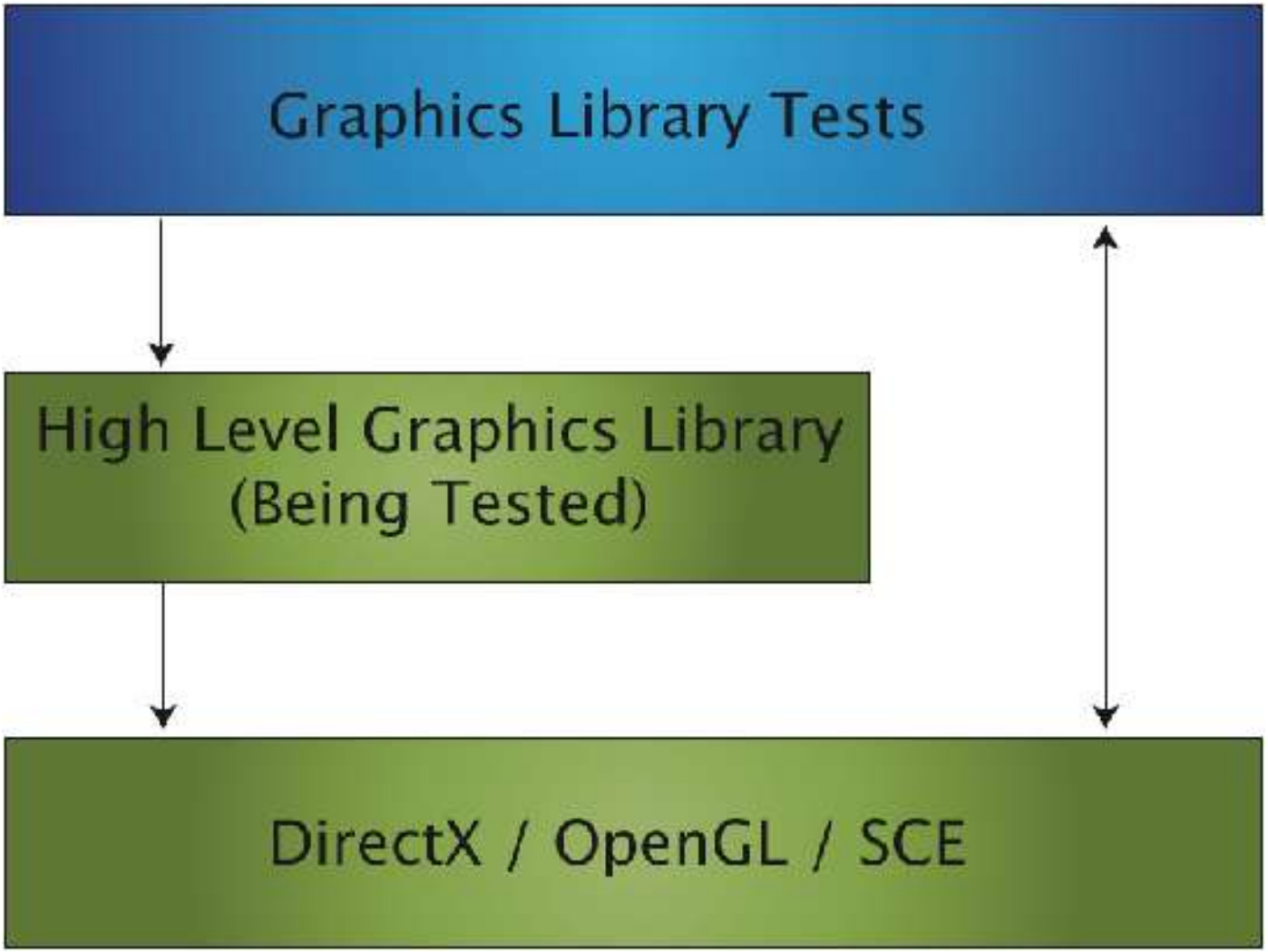
Run the tests on consoles... less often


CruiseControl.NET - Mozilla Firefox
CruiseControl.NET
Latest
Next
Previous
Build Report
View Build Log
NAnt Output
NAnt Timings
Recent Builds
2006-03-20 16:52:48 (CCnet-Ue3-PC-Dbg-1771)
2006-03-20 16:34:35 (CCnet-Ue3-PC-Dbg-1770)
2006-03-20 16:20:33 (CCnet-Ue3-PC-Dbg-1769)
2006-03-20 15:36:11 (Failed)
2006-03-20 15:15:06 (Failed)
2006-03-20 15:07:07 (Failed)
2006-03-20 14:58:35 (Failed)
2006-03-20 14:34:58 (CCnet-Ue3-PC-Dbg-1768)
2006-03-20 14:06:10
BUILD FAILED
Project: UE3 Incr. Build (PC-Debug)
Date of build: 3/17/2006 4:08:14 PM
Running time: 00:19:26
Build condition: Modifications Detected
Last changed: 2006-03-17 16:01:58
Last log entry: - Added descriminator for idle-to-walk left and right
Added turn quadrant enum and selector
Errors: (9)
Warnings: (1)
LINK : warning LNK4098: defaultlib 'MSVCRTD' conflicts with use of other libs; us
Unit Test Summary (C++ and Script)
Tests Run:
2055 test(s) run
ScriptLog: UnitTest: 2348 test(s) run

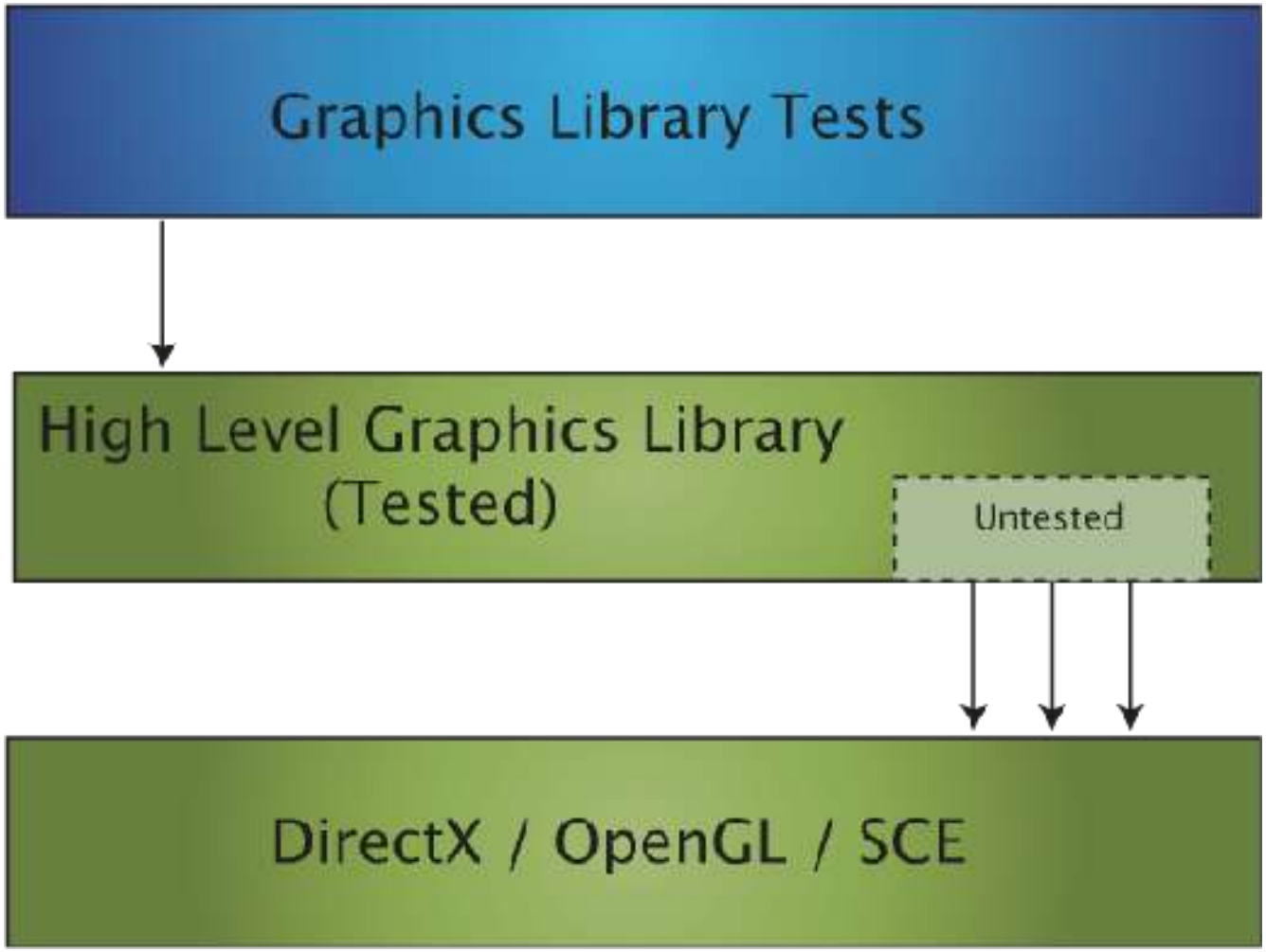
Wrap full API



Test API state directly



Test all code except for API calls

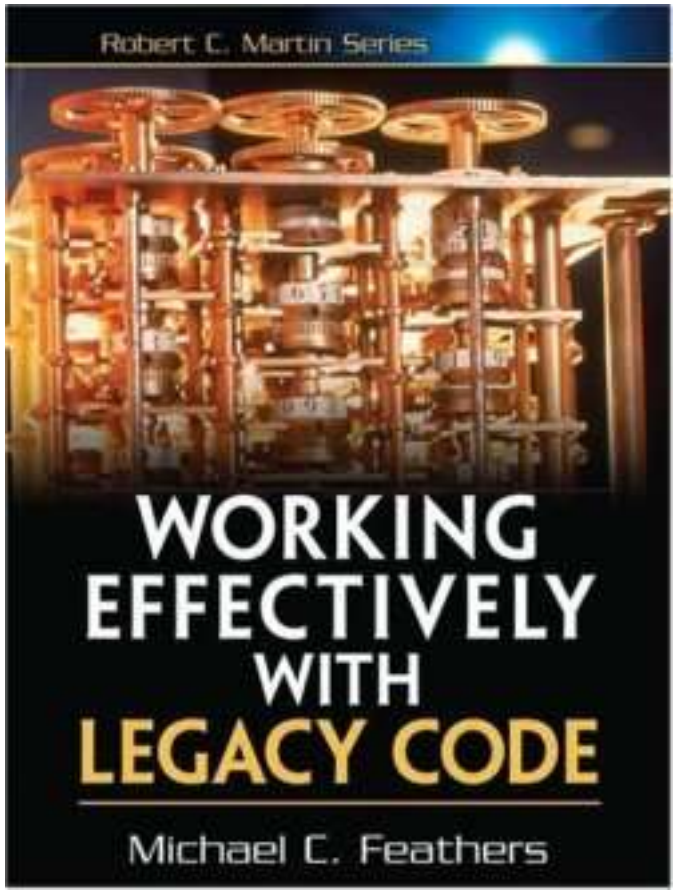


Testing With Middleware

Havok
RenderWare
Unreal
Novodex
OpenGL
DirectX

TDD With An Existing Engine

I'd like to use TDD but...



~~1. What is TDD?~~

~~2. How We Use TDD~~

~~3. TDD and Games~~

4. Lessons Learned

5. Wrap Up

Lesson #1: TDD can be used for high-level game code

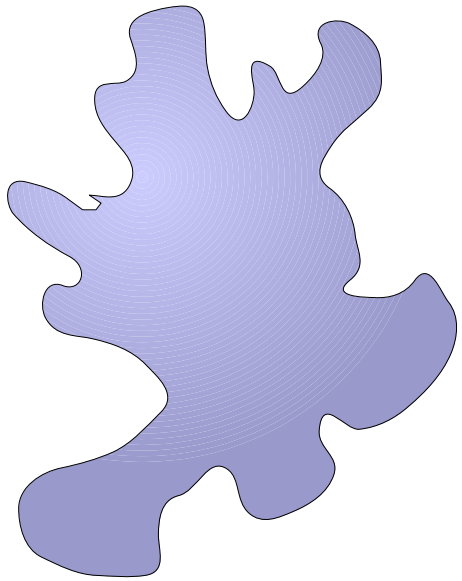
Fighting AI example

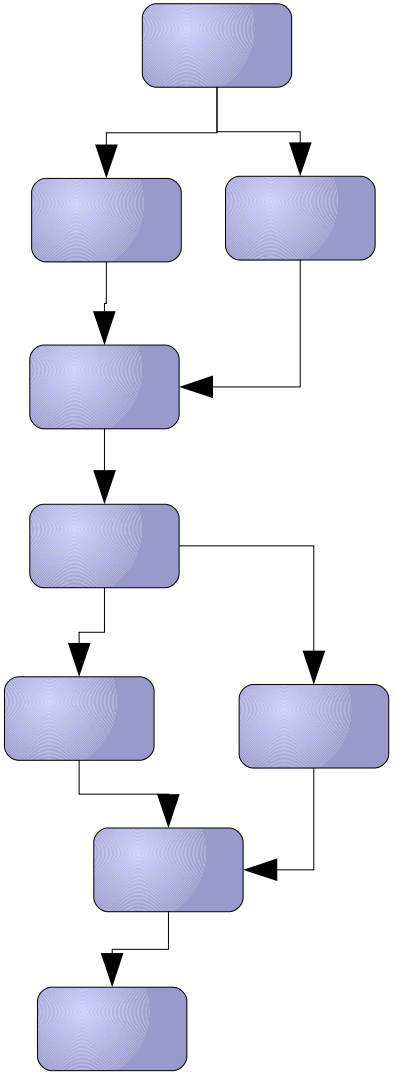
```
function TestEnemyChoosesLightAttack()
{
    FightingComp = new(self) class'FightingComponent';
    FightingComp.AddAttack(LightAttack);
    FightingComp.AddAttack(HeavyAttack);

    enemy.AttachComponent(FightingComp);
    enemy.FightingComponent = FightingComp;
    enemy.FindPlayerPawn = MockFindPlayerPawn;
    enemy.ShouldMeleeAttack = MockShouldAttack;
    ShouldMeleeAttackReturn = true;

    enemy.Tick(0.666);

    CheckObjectsEqual(LightAttack,
        FightingComp.GetCurrentAttack());
}
```

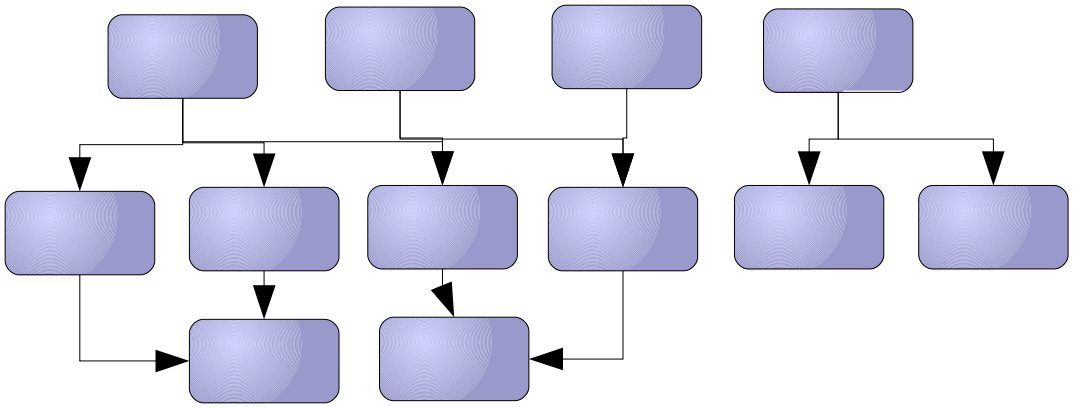
Character behavior example

```
TEST_F( CharacterFixture,
        SupportedWhenLeapAnimationEndsTransitionsRunning )
{
    LandingState state(CharacterStateParameters(&character),
        AnimationIndex::LeapLanding);

    state.Enter(input);
    input.deltaTime = character.GetAnimationDuration(
            AnimationIndex::LeapLanding ) + kEpsilon;

    character.supported = true;
    CharacterStateOutput output = state.Update( input );
    CHECK_EQUAL(std::string("TransitionState"),
            output.nextState->GetClassInfo().GetName());
    const TransitionState& transition = *output.nextState;
    CHECK_EQUAL(std::string("RunningState"),
            transition.endState->GetClassInfo().GetName());
}
```

Choice of architecture

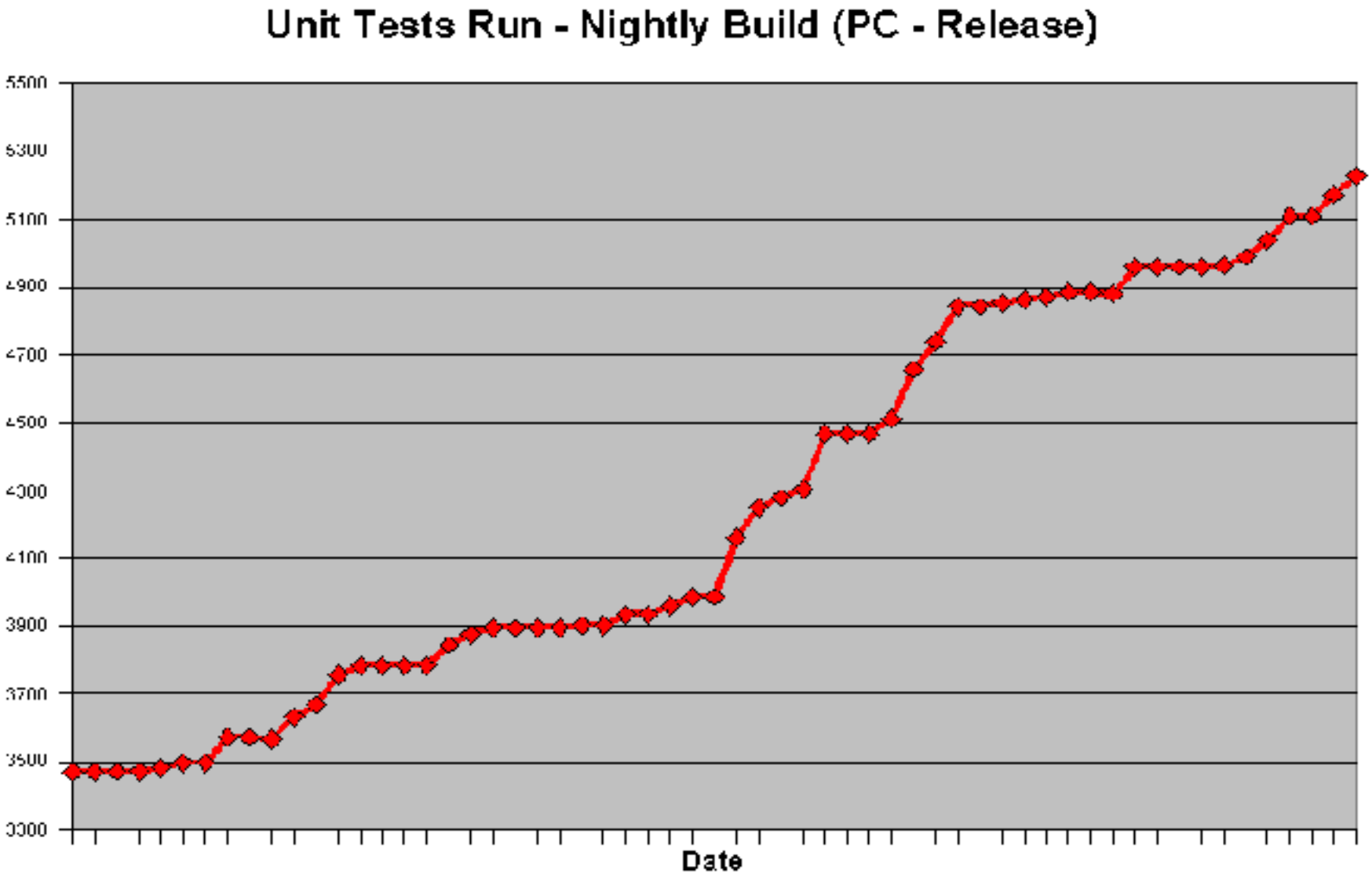






Lesson #2: TDD and code design

Lesson #3: Number of tests as a measure of progress

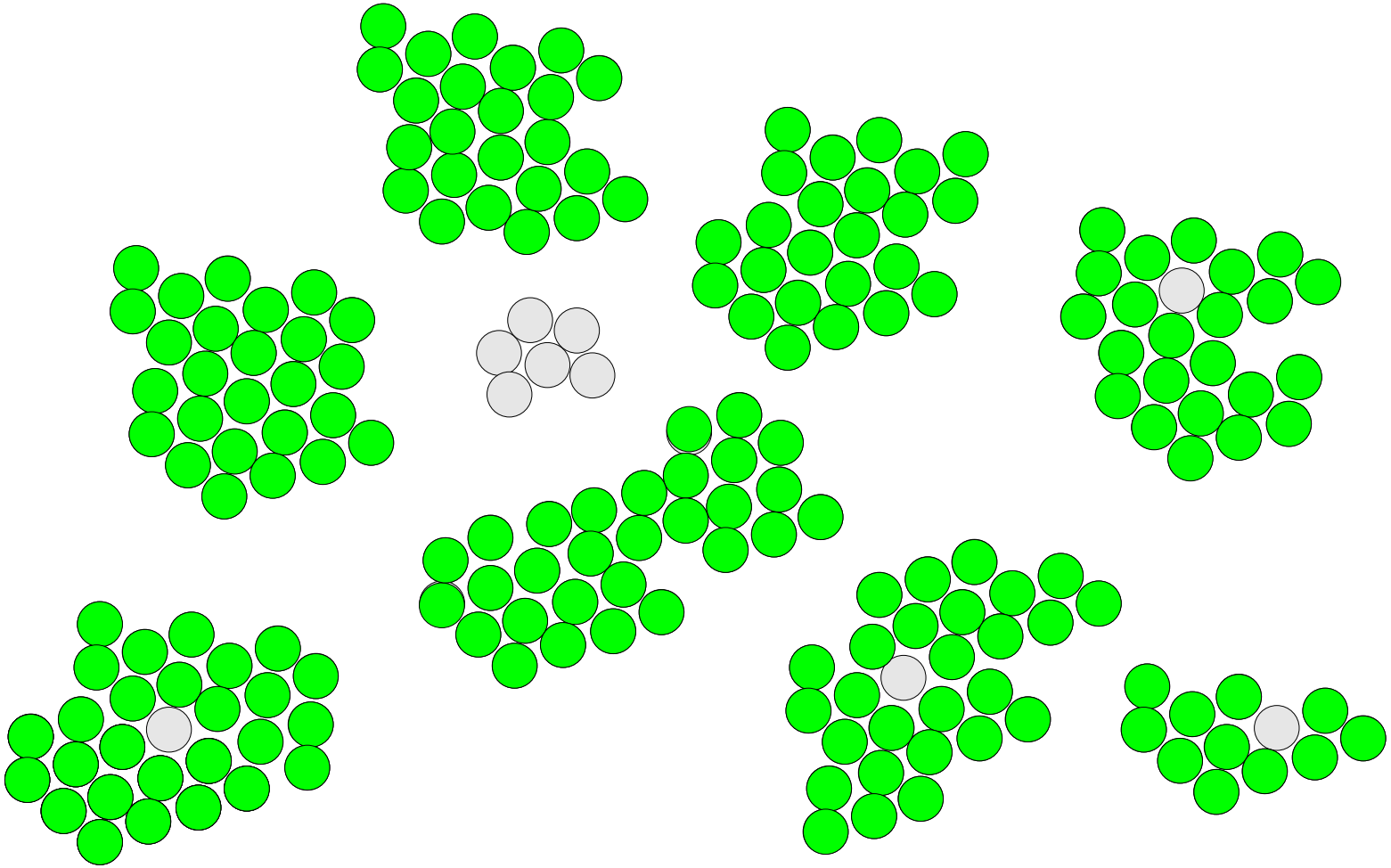


Lesson #4: TDD improves build stability

Lesson #5: TDD creates more code

Lesson #6: Development speed

Lesson #7: Adopting TDD



~~1. What is TDD?~~

~~2. How We Use TDD~~

~~3. TDD and Games~~

~~4. Lessons Learned~~

5. Wrap Up

Conclusion

Resources

Games from Within http://www.gamesfromwithin.com

Includes paper for this presentation with more details and links to other TDD resources.

Noel Llopis - llopis@convexhull.com

Sean Houghton - sean.houghton@gmail.com

Questions?

GameDevelopers Conference