게임 프로젝트 관리: 게임 QA

QA실질 업무: 프로젝트 산정, 게임 개선 방안 도출

QA(Quality Assurance): 품질 보증

**품질 수행 코칭(프로세스 적용 가이드 등)**

**프로세스 품질(프로세스 준수 Audit)**

**프로덕트 품질(테스팅, 지표 수집)**

소프트웨어 개발의 전 영역에 참여하여 SW의 복합적인 품질 요소에 대한 보장성을 확보하는 일련의 과정

적합성 체크: 산출물과 프로세스를 표준에 의거하여 검사, 체크리스트 사용

품질 수준 평가: 산출물의 내용물, 정확성, 적합성을 검토, 테스트를 통해 확인

품질 수행 능력 코칭: 적절한 프로세스를 어떻게 선택하여 사용하는 지 자문, 프로젝트 특성에 맞게 테일러링 가이드

게임 QA의 필요성

일반 제품

* 제품은 팔기 위해 존재
* 소비자들의 많은 부류가 이미 구입하고 나서 잘 못 샀다고 판단
* 적극적인 반품에 대한 수요 비율이 낮다
* 이후에 이 제품을 사지 않겠다고 판단하지만 제품은 이미 팔림

게임

* 게임은 고객이 한 번의 선택으로 구매가 되는 것이 아님
* 처음부터 구매가 이루어지지 않지만 게임에 대한 평가는 지속적으로 진행
* 게임에 대한 평가가 떨어질 경우 이후 접속 유도가 매우 힘들다

요즘 모바일 게임 시장: 좋은 게임 엔진들이 많이 보급되면서 어느 회사든 빠르게 만들 수 있는 환경 조성 🡪 시장에 빠르게 진입하는 게 중요(프로젝트를 최대한 빠르게 수행하는 게 중요) = 개발에만 전념해야 된다(무슨 일을 얼마만큼, 어느 정도의 수준으로 해야 되는지에 대한 고민을 할 시간이 없다)

게임 QA 기준

절대적 기준: 목표에 따른 내부적 요소

* 게임 콘텐츠 수량
* 게임의 타겟 층/타겟에 따른 적절한 콘텐츠
* 목표 된 플레이 시간(콘텐츠 소모량)
* 내재하고 런칭할 문제점의 수위

상대적 기준: 경쟁 제품 대비 우위 또는 동급에 있어야 할 요소

* 게임 내 밸런스
* 편의적 사항
* 하드웨어 호환성
* 네트워크 사용량/네트워크 성능

이게 기획 의도와 예상 효과가 타당한 가에 대한 것도 검토해야 🡪 분석 역량도 중요

게임 QA의 필요 요소

1. 개발 방법론 및 내부 프로세스 표준화 방안
2. 품질 인식 및 QA 업무 진행 매뉴얼
3. 개발 진행 정보 및 개발 솔루션
4. 제품 분석 및 시장 분석, 트렌드 조사
5. 개발 진행 히스토리 및 앞으로의 목표
6. 개발 콘셉 및 개발 리소스
7. 목표 된 일정과 개발 리스크

QA의 역할

예시) 넷마블 QA 팀

QA

* 테스트 전략 및 방향 결정 QA 계획 수립
* QA 수행에 필요한 적정 인력 산정 및 충원 요청
* BTS(Bug Tracking System) 세팅, 테스트 계정 생성 등 QA 활동에 필요한 테스트 환경 구축
* 빌드 별 테스트 수행 상황을 모니터링하고, BTS에 등록된 버그 확인 및 테스트 단계 별 QA status 결과를 유관 부서에 공유
* 테스트 결과를 토대로 Sign – off 기준 달성 여부 판단, 유관 부서와 오픈 적합 여부 판단
* 개발사 또는 사업부로부터 전달받은 게임 기획서(또는 수정된 기획서)등 테스트에 필요한 모든 정보를 공유하고, 테스트케이스 작성을 가이드
* 해당 게임의 보안 검수 및 보안 툴 적용을 챙길 책임
* QA 기간 동안 테스트 진행을 포함한 QA 업무 수행 전반이 QA 계획대로 진행되도록 코디 및 유관 부서, 개발사와의 의사소통과 의사 결정 수행
* 게임에서 발생한 장애의 조치 결과 확인 및 장애 보고서 작성을 통한 사후 관리
* 테스트 지원이 활발하게 이루어질 수 있도록 인력을 관리하고 TL/TE를 교육

IGS QA팀

TL (Test Leader)

* QA 계획에 따라 테스트를 설계하고, TE를 통해 테스트를 수행
* 테스트 결과의 신뢰성을 보장할 수 있도록 TE의 각종 산출물을 검토
* 테스트 수행 결과를 정리하여 QA담당에게 보고
* 테스트 진행 상황을 점검하고 문제 발생 시 문제 해결을 위한 의사소통 수행
* 등록된 버그를 검토, 우선 순위를 결정하여 개발사에 버그 수정 의뢰하고, 종료 시까지 추적 관리 및 보고(테스트 진행 중 중요한 이슈 발생 시 QA담당에게 즉각 공유할 책임)
* 장애 발생 시 QA요청에 따라 장애 복구 지원

TE (Test Engineer)

* QA 담당의 QA 진행과 TL의 테스트 설계에 따라 테스트 수행
* 테스트 요구 사항과 기획서를 분석하여 테스트케이스를 도출하고, 테스트케이스에 기반하여 테스트 수행 후 결과 보고
* 사용자 시나리오를 기반으로 플레이 테스트를 수행하여 결과 보고
* 테스트에서 발견된 버그를 BTS에 등록하고 종료 시까지 추적 관리
* 장애 발생 시 QA 요청에 따라 장애 복구 지원

QA 시점

1. 알파, 클로즈 베타, FGT같은 특정 시점
2. 서비스 오픈 전
3. 서비스 오픈 후
4. 점검(임시점검/정기점검)
5. 장애 복구

버그 관리

버그 관리 순서

* 결함 등록(new) 🡪 담당자 할당 🡪 버그 확인/수정 🡪 수정 확인 후 종료
* 재발견(Reopen) 🡪 담당자 할당 🡪 버그 확인/수정 🡪 수정 확인 후 종료

버그로 체크할 수 있는 목록

1. 명세서 대로 기능이 작동하지 않음

* 상점에서 한 번에 여러 개의 아이템을 구입할 수 있어야 하는 데 한 번에 한 개씩 밖에 살 수 없다

1. 명세서에 없는 방식으로 기능이 수행됨

* 전설 급 아이템은 상점에서 팔면 안 되는데 팔고 있다

1. 명세서에는 없는데 필수적인 기능이 제대로 수행되지 않음

* 유저가 돈을 100원만 들고 있을 때 100원을 초과하는 아이템을 사려고 하면 골드 충전 팝업이 바로 떠야 되는데?

1. 명세서 대로 작동되기는 한데 유저의 관점에서 사용이 명백히 힘듦

* 현재 여러 개의 아이템을 구입할 때 슬라이드 바로 수량 조절이 되지만 정확한 수량 조절을 위한 미세 조절 기능(좌우 스크롤 버튼이나 숫자 직접 입력 하는 공간)이 필요하다

결함 등록 시 우선 순위, 중요도의 기준

|  |  |  |
| --- | --- | --- |
| 항목명 | 값 | 설명 |
| Priority  (우선 순위) | Low(P4) | 가장 나중에 처리해도 되는 문제, 발생 빈도가 낮거나, 잘 재현 되지도 않고, 재현 가능성 낮은 경우, 이용자가 낮은 서비스에서 발생하는 문제 |
| Normal(P3) | 일반적인 기능이 바르게 동작하지 않아 사용자의 이용이 원활하지 않으나, 다른 대체 방안이 있는 경우 |
| High(P2) | 이용도가 높은 서비스나 기획 이슈가 높은 기능에 대해 바르게 작동하지 않는 경우 |
| Urgent(P1) | 가장 먼저 처리해야 되는 우선 순위가 높은 결함, 서비스 진입이 전혀 안 되는 결함, 잘못된 노출로 인해 사용자에게 물리적, 금전적 피해를 주는 결함 |
| Severity  (심각성) | Trivial(S4) | 기능 상 중요하지 않고, 고객에게 피해를 주지 않는 사소한 기능이나 디자인 문제 |
| Minor(S3) | 일반적인 기능 오류로 수정이 필요한 문제 |
| Major(S2) | 일부 기능에 대해 테스트 진행이 안 되거나, 주요 기능에 문제가 발생하거나, 서버 에러가 발생하여 주요 기능에 영향을 주는 문제, 성능 저하로 정상적인 이용이 불가능한 문제 |
| Critical(S1) | 테스트 진행을 막히게 하거나, 사용자 데이터가 손실되거나, 시스템이 다운되는 문제 |
| 보안 문제가 있거나 결제 진행이 문제가 있는 경우 |

면접 기출: 애니팡에서 발생할 수 있을 법한 결함 예시 🡪 해결 우선 순위 결정해 봐라

친구 기능이 안 됨/ 닫기 버튼이 작동 안 함/ 하트 충전이 안 됨 🡪 하트 충전 문제가 최우선

**돈, 사용자 데이터 관련된 이슈는 무조건 1순위!**

테스트 환경 구축

OS 종류에 따른 분류

하드웨어에 따른 분류

* PC: CPU, 그래픽 카드 등
* 모바일: iOS, Android(갤럭시, G시리즈, 베가 시리즈 등)

네트워크 환경에 따른 분류(Wi-fi, 3G, LTE)

소프트웨어 빌드 테스트 용어

단위 테스트 (Unit Test)

* 개발자가 자신이 작성한 소스 코드를 테스트하는 것
* 비공식적으로 진행, 진행 단위는 케이스에 따라 다름

시스템 테스트 (System Test)

* 결함을 찾아 내기 위해, 소프트웨어를 실행하여 테스트를 진행하는 것을 의미
* 테스트 조직에서 담당하며, 요구된 사항을 토대로 테스트 계획서를 작성하여 케이스를 만듦
* 예시: 5천명이 들어갈 수 있는 서버에서 플레이할 거야 🡪 5천명 분의 더미 패킷들을 만들어서 실제 플레이어가 들어온 것처럼 테스트

스모크 테스트 (Smoke Test)

* 새로운 빌드를 만든 후, 테스트를 진행하기에 앞서, 해당 빌드가 테스트 받을 만한 가치가 있는 것인지 검증하는 과정
* 빠르게 테스트를 할 수 있는 형태로 만들어서 진행하는 검사

소프트웨어 빌드 테스트 용어

빌드 검증 테스트(BVT: Build Verification Test)

* 빌드 전반에 대한 내용을 테스트하는 것이 핵심
* 빠르게 빌드 전반에 대한 내용을 테스트하는 게 중요
* BVT기준 참조

1. 모든 것을 자동화: 설치 삭제 자동화 등 테스트를 효율적으로 진행
2. 일부만 테스트: 기본 기능을 확인하여 테스트 진행 여부 결정
3. 신속하게 테스트: 수행 시간이 짧을수록 빌드의 문제를 즉각 파악 가능
4. 실패를 정확히 인지: BVT가 실패했다면, 실패 원인은 즉시 수정
5. 깊게가 아닌 넓게: 세세한 부분은 나중에 하고 주요 기능, 주요 사용 시나리오를 가능한 많이 포함
6. 디버그와 유지 보수가 용이: 실패가 발생하면 그 원인을 파악하여 목록화
7. 신뢰할 수 있어야: 실패 발생 즉시 알려야 한다. 타협 금지

빌드 수용 테스트(BAT: Build Acceptance Test)

* BVT와 같은 것이지만 BVT가 완료된 빌드를 QA가 테스트하는 것을 BAT

회귀 테스트(Regression Test)

* 소프트웨어를 수정한 후, 과거에 고쳤던 버그가 다시 살아나는 것을 회귀 버그(Regression Bug)라고 하고 그 버그를 찾는 테스트
* 이 문제를 해결했더니 다른 부분에는 영향을 주지 않는구나를 밝혀내는 테스트

프로그램/리소스 🡪 빌드 🡪 실행 파일 리소스(이 과정에서 BVT) 🡪 BAT 🡪 클라이언트(사용자 평가)

일반적인 테스팅 방식

1. 빌드 확인 테스트(BVT/BAT)

* 새 빌드가 세부 단계의 테스트를 진행할 수 있을 만큼의 완성도가 있는지 빠른 시간 내에 검증하는 테스트

1. Positive Test

* 정상적인 프로세스를 진행했을 때 정상적인 결과가 나오는지 테스트

1. Negative Test

* 비정상적인 프로세스로 진행했을 때의 결과를 테스트

1. 비 기능성 테스트

* 기능 이외의 실제 동작 항목에 대한 테스트 (호환성/속도/스트레스 테스트 등)
* 예를 들면 인터넷 속도를 겁나 낮춰버린다 거나

게임 테스트

1. 기능성 테스트 – 게임 내의 기능들이 정상적으로 작동하는 지
2. 호환성 테스트 – 하드웨어의 변화, OS의 변화, 사양의 변화에 따라 정상적으로 기능이 작동하는지
3. 메모리 누수 테스트 – 게임 프로그램을 사용하면서 메모리 사용량이 증가하는 상황이 발생하는 지
4. 게임 학습 테스트 – 게임 플레이 학습이 정상적으로 이루어지는 지/ 게임 플레이 학습하기 쉬운 구조가 갖춰줬는지
5. UI 테스트 – UI의 기능이나 정보의 치환성이 정상적으로 잘 이루어지고 있는 지
6. 퍼포먼스 테스트 – 클라이언트의 퍼포먼스, 프레임, 컴퓨터 사양의 역량 등에 따른 테스트/ 얼마까지 퍼포먼스를 끌어올릴 것인 지, 1개의 데이터 베이스 당 어느 정도의 데이터를 수용할 수 있는가
7. 설치/제거/패치 테스트 – 설치 과정에 문제가 없는지, 제거할 때 문제가 없는지, 패치 내역 적용에 이상이 없는 지
8. 서버 스트레스 테스트 – 대역 폭은 어느 정도까지 버틸 수 있는지/ 최대 얼마까지 수용 가능한 지
9. 재미 테스트 – 게임의 재미 측면에서 테스트(콘텐츠 소모 시간, 의도했던 재미 요소들이 잘 반영 되었나, 요구 분석, 사례 분석): 게임의 재미 산정을 위해 어떤 지표를 사용할 수 있을 것인가

TC(Test Case): 테스트 케이스

체크리스트를 바탕으로 테스트 커버리지를 강화하기 위해 작성된 테스트 항목

TC제작은 단순 테스터와 QA를 구분하는 가장 중요한 업무 항목

업무 경험/게임에 대한 지식과 경험/ 게임 프로젝트 진행/ 각 파트의 업무에 대한 이해도 필요

테스트 케이스 작성 예시

목적: 캐릭터 명 생성 부분에 대한 테스트

Test Basis 확인

* 캐릭터 명은 최소 영문 4자/ 한글 2자부터 최대 영문 8자/ 한글4자 까지 입력 가능
* 영문, 한글, 숫자를 제외한 특수문자(공백 포함)등 다른 문자를 사용할 수 없다

Positive Test 🡪 단위 테스트(끝에서 끝의 값만 검사)/ 결제 시스템의 경우에는 모든 값에 대해서 검사를 진행해줘야 한다

* 영문 4자로 캐릭터 명 생성 시도
* 한글 2자로 캐릭터 명 생성 시도
* 영문 8자로 캐릭터 명 생성 시도
* 한글 4자로 캐릭터 명 생성 시도

Negative Test

* 영문 4자/ 한글 2자 미만으로 캐릭터 명 생성 시도
* 영문 8자/ 한글 4자 초과한 캐릭터 명 생성 시도
* 문자의 맨 앞, 중간, 맨 뒤에 공백 문자 삽입 시도
* 특수문자 입력 시도(특히 사칙연산기호, ;<>{}등 소스 코드나 스크립트 문법 관련 기호 위주)
* 중복된 캐릭터 명 입력 시도
* 금지 어 입력 시도(비속어, 게임 내 NPC명, ‘운영자’ 등)

체크리스트는 명세를 통해서, 테스터의 역량을 통해서 테스트 진행: 테스터의 역량에 따라서 정확도가 떨어질 수 있다(목표 지향 방식)

테스트 케이스는 세세하게 지정해서 테스트를 진행하는 사람이 어떻게 진행해야 되는 지까지 지정해서 진행: 내가 원하는 테스트 결과를 정확히 도출할 수 있다/ 어떻게 테스트하면 버그가 발생하는 지 파악 가능(절차 지향 방식)

테스트 케이스는 QA역량이 전혀 없는 사람에게 줬을 때에도 그 사람이 지장없이 테스트에 임할 수 있을 정도의 수준까지 만들어야 됨