퀘스트 기본

시작 조건에 어떤 것들이 필요한가 🡪 속성, 기능이 어떤 것들이 있어야 할까?

스토리를 전달하는 방법

어떤 플레이를 할 수 있게 할까?

보상에 해당하는 기능과 속성은 무엇이 있지?

퀘스트의 역할

튜토리얼 및 초반

초반 유저에 대한 전반적인 가이드 역할이 가장 중요

게임의 세계관 및 스토리에 대한 **정보 제공**

핵심적이고 기본적인 게임 플레이에 대한 **학습**

유저의 **호기심**과 **궁금증**을 자극

유저에게 게임 플레이에 대한 **동기 부여**

튜토리얼 설계에 따라 게임의 첫 인상 결정(가장 어렵고 가장 중요하다) 🡪 마케팅 쪽에서 활용되는 사례도 많다

처음 스테이지에 이야기들을 가지고서 한 단계 한 단계 플레이를 학습을 할 수 있게끔 만들 것이냐에 대한 고민이 이루어져야

메인 퀘스트

가장 중요한 목적: **콘텐츠 소모 속도 조절**

**메인 스토리**와 **세계관**에 대한 학습 및 몰입감 향상

유저의 전체 플레이 동선으로 활용

플레이어의 다음 게임 플레이에 대한 **가이드**

* 새로운 지역 오픈, 시스템 및 콘텐츠 소개 등

**유저 잔류, 재 접속, 결제** 등을 유도하는 장비, 아이템, 스킬 등 지급

서브 퀘스트

다양한 성향(플레이 패턴, 직업, 레벨 등)의 유저가 사용 가능한 콘텐츠의 양적 제공

게임 내 활용성이 낮은 리소스의 활용(맵, 몬스터 등)

직업 별 밸런스 조절

독립된 시즌 성 퀘스트: 크리스마스, 할로윈, 올림픽 금메달 기념 이벤트 등

게임 플레이에 따른 퀘스트 배치

게임 플레이 후반으로 갈수록 발생하는 퀘스트의 변화

* 퀘스트 클리어 타임 상승: 난이도 상승
* 퀘스트 스케일 확대: 개인적 문제 🡪 지역적 문제 🡪 세계적 문제
* 퀘스트 보상 가치가 강력 해진다

개인 퀘스트는 퀘스트보다 장소 이동이 빈번하다

* 메인 퀘스트는 주를 이루기 때문

팀 퀘스트는 개인 퀘스트보다 시도하기 어렵기 때문에 **더 좋은 보상**이 주어져야 한다

파티 퀘스트에서 이동이 빈번하면 안 되는 이유: 이동이 빈번하다는 것 = 중간 이탈 요인이 많아서 플레이 몰입이 깨지게 됨 🡪 반복될 시 플레이어 이탈 요인 중 하나가 됨

이동이 많은 방식보다는 한 두개의 맵에서 집중해서 플레이할 수 있는 형태를 구성해야

버려지는 클래스가 존재하지 않게 다양한 조합을 유도할 수 있어야 한다

퀘스트 디자인 기초

퀘스트 디자인 순서

1. 제작 의도 설정
2. 기본 스토리 라인 설정
3. 시작 조건 설정
4. 진행 설정
5. 보상 설정

제작 의도 설정

해당 퀘스트를 통해 유저에게 제공하고자 하는 것

유저로부터 얻고자 하는 효과

* 예: 산성 슬라임은 산성 분비물이 무기 내구성을 낮추기 때문에 유저들이 사냥을 기피 🡪 산성 슬라임 20마리를 잡으면 무기 아이템을 준다

즉석 질문

악마의 동굴은 악마의 동굴 지하의 악마를 잡으면 랜덤 아이템을 드랍하는 **반복 퀘스트**다. 제작 의도는? 🡪 플레이어가 원하는 장비를 얻을 때까지 플레이 유도 🡪 플레이 타임 확보 목적

(확률에 대한 고민, 보상 산정 등 고민해야)

제작 의도 예시

새 시스템 소개

지역 탐색

메인 스토리 학습 유도

특정 몬스터 사냥 유도

기본 스토리라인 설정

세계관 또는 설정된 시나리오를 기반으로 설정

해당 관계의 퀘스트에 적합한 형태의 스토리라인 선택

해당 퀘스트의 스토리 적인 발단 동기 설정

해당 퀘스트의 스토리 적인 진행 및 마무리 설정

해당 퀘스트 완료 후 게임의 세계관 및 시나리오에 미치는 영향 설정

시작 조건 설정

퀘스트가 발동되거나 시작할 수 있는 조건을 설정

조건이 잘 못되었을 경우 발생하는 문제점

* 퀘스트가 시작이 안 되요, 저렙인데 고렙 퀘를 받을 수 있네? 전사만 받을 수 있는 퀘스트를 마법사가?

캐릭터의 상태

* 직업, 레벨, 아이템, 명성 등

타 퀘스트 선행 여부

기타 특정 조건 만족

* 밤에만 받을 수 있거나 평판이 ‘매우 선함’ 일 때만 받을 수 있다 등

진행 설정

1. 진행 순서 🡪 예를 들면 주점에 가면 자동으로 퀘스트가 진행되는지 아니면 선장과 시비가 붙은 주정뱅이들을 떼어낸 다음에 진행이 되는지
2. 진행 루트 🡪 이동 동선은 어떻게 되는 지
3. 수행해야 하는 임무 🡪 사냥을 해야 되는지, 보스를 잡아야 되는지, 재료를 모아 오던지
4. 난이도 🡪 레벨 구간에 맞게 사냥을 하게 할 지, 파티를 구성해야 할 지, 아이템 확보에 대한 난이도나 지역 탐색에 관련된 난이도를 부여할 건지
5. 시간 등 🡪 퀘스트 수락과 퀘스트 수행 지역까지 이동, 임무 수행, 복귀까지 시간을 합쳐서 퀘스트 하는 데 걸리는 시간 책정 🡪 시간 대비 보상 판단 가능

퀘스트 임무 종류

1. 대화
2. 아이템 배달
3. 지역 방문, 지역 탐색
4. 사냥
5. 수집
6. 구출
7. 호위
8. 보스 몬스터 처치
9. 오브젝트 파괴
10. 특정 스킬 사용 🡪 튜토리얼에 사용하면 좋은 모델
11. 타임 어택
12. 특정 콘텐츠 사용 강제 🡪 예: 투기장에서 몇 승 이상하고 와라
13. 장애물 돌파

이외에도 기획자의 창의성을 발휘하여 구성 가능

보상 설정

재화 적 측면: 돈, 아이템 🡪 직업에 따른 아이템 구분 값, 몇 개를 줄 지

성장 적 측면: 경험치, 명성, 스킬 🡪 직업에 따른 아이템, 스킬 구분

기능 적 측면: 입장 권한, 타 퀘스트 실행 조건, 메뉴 확장 등 🡪 A퀘스트를 클리어하면 강화 메뉴를 띄워주고 사용할 수 있게 해줘라(이 퀘스트가 몇 번이고 뭐가 열리고 어디로 갈 수 있다 정의 필요)

좋은 진행과 보상 설계

동일한 시간의 사냥에 대한 보상을 기준으로 설계

사냥 등 정상적인 플레이로 얻을 수 없는 의미 있는 결과물

퀘스트 수행 시간이 길수록, 난이도가 높을수록 더 강력한 보상

* 비슷한 보상을 주거나 비슷한 상태에서 수행 가능한 타 퀘스트 대비 동선이 길거나, 난이도가 높으면 외면당할 가능성이 높다

유저의 속성에 대한 파악해야 한다

1. 유저들은 유능감을 과시하기를 원하는 지, 조금 더 똑똑한 플레이를 하고 싶은 지
2. 적당히 대충하고 넘어가고 싶어한다

연계 퀘스트

수 많은 퀘스트 들의 관리를 용이하게 한다

단일 퀘스트가 가진 문제들을 다른 퀘스트와 연계하여 해결

* 재미 부족, 적나라한 목적성, 노력과 보상의 불일치 등

사냥, 배당, 아이템 수집, 수색 등 다양한 목적을 가진 퀘스트 들을 묶는 것이 중요

* 가장 마지막 퀘스트는 보상 수준이 강력해야 한다

전체 퀘스트 구성

게임 전체의 퀘스트(혹은 한 시즌 전체)는 밸런스 측면을 고려해서 구성

레벨, 스토리의 진행과 함께 **퀘스트의 연계 구조**를 표시하는 한 장의 **다이어 그램**

각 퀘스트의 타입, 사용 지역, 몬스터, 주요 보상, 싱글 or 멀티, 수행 클래스, 플레이타임 등을 정리한 **시트**

위의 시트를 통해 전체 퀘스트가 게임의 각 리소스, 타입, 클래스 등을 얼만큼 활용하고 있는지를 한 눈에 리뷰할 수 있는 **요약 페이지**

퀘스트 구조 설계

초 간단 퀘스트 구조

유한상태기계(FSM) + 스위치 + 트리거 조합으로 구성 가능

초기 RPG에서 많이 쓰임

장점: 테이블화를 통해 퀘스트의 대량 제작, 양산, 수정에 용이

단점: 구조적 한계로 표현에 제약이 많다. 유저가 게임 세계에 몰입할 수 있는 스토리텔링 전달이 어렵다

FSM: 한 개의 객체가 여러 개(유한)의 상태 가질 수 있는 객체들이 어떻게 상태가 전이되는지 정의

특정 상태가 특정 사건에 의해 다른 상태로 전이하는 구조

객체에 True, False를 체크하는 속성 추가

트리거

조건과 작용 및 자주 사용하는 구문을 하나의 단위로 묶어 놓은 것

조건과 작용: 트리거의 기본 구조: 예시 – ‘A조건을 만족하면 A’작용이 실행된다

예시: 퀘스트를 만들어보자

조건: 장로에게 말을 걸어라

작용: 장로가 기본 아이템을 준다

* 위 이벤트는 저 레벨 구간 유저들에게만 적용되게 하는 게 좋을 듯 (저 레벨 구간에 대한 기준 정의, 최대 레벨 제한 값 정의)
* 위 구조대로 설계하면 유저는 장로에게 말을 걸 때마다 아이템을 받게 되지 않나?
* 유한 상태 기계를 통해 유저가 퀘스트를 받을 수 있는지 없는지, 퀘스트를 수행했는 지에 대한 확인 필요(완료 체크 필요)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | maxLV | 행동 | 대상 | 대사 | 작용 | 대상 | 완료 |
| 1 | 5 | 대화 | 장로 | 대사ID | 획득 | 기본아이템 | Bool |

퀘스트 하나 만들어 보기

퀘스트 받을 수 있는 상태(A): 스위치 off

퀘스트 종료 = 스위치 off

A상태의 전이 사건1: 레벨 5이하

A상태의 전이 사건2: 기본 아이템을 갖고 있지 않을 때

조건: 장로에게 말을 건다

작용1: 장로가 기본 아이템을 준다

작용2: 퀘스트 종료 스위치 On

연속되는 퀘스트 설계

장로에게 말을 걸면 기본 아이템을 주고 대장장이에게 가 보라고 하는 퀘스트가 생성되는 경우

앞의 퀘스트 작용에 아래의 작용을 추가

작용: 2번 퀘스트를 받을 수 있는 상태 스위치 = On(해당 스위치를 대장장이 퀘스트에 연결하면 된다)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 연결 퀘스트 ID | 선행 퀘스트 ID | 행동 | 대상 | 대사 |  |  |
|  | 완료 참조 |  |  |  |  |  |

스위치의 확장

스위치 값 필드를 퀘스트 트리거뿐 만 아니라 몬스터, 아이템, NPC CLASS에 모두 삽입하는 방식으로 퀘스트 관리를 용이하게 한다

* 예: 평소에는 고블린이 깃발 아이템을 드랍하지 않지만, 고블린 부족 깃발을 10개를 모아오라는 퀘스트 수락 이후에는 고블린을 죽이면 깃발 아이템 드랍

스위치를 통해 맵, NPC, 스토리 진행 등 게임 세계의 상태를 콘트롤 한다

* 생성, 소멸, 오픈, 특정 액션을 취함, 선택 가능한 기능의 증가
* 하지만, 스위치의 확장 사용부터는 실질적으로 Class 설계만으로는 몰입도 높은 퀘스트 구현이 힘들어 짐. 스크립트 언어의 적극적인 사용 필요

퀘스트 Class 설계

각 퀘스트 별로 ID 부여

각 퀘스트에 대응하는 트리거와 스위치를 할당

시작 조건 리스트 제작

* 레벨, NPC 특정 아이템 소유 등

작용 리스트 제작

* 아이템 주기, 아이템 가져가기, 경험치, 돈, 스킬 포인트, 우호도 보상 등

스위치 확장을 이용한 NPC, 혹은 월드 전체의 상태 변화 적용

* 임의의 퀘스트를 서술형 형태로 작성해 보고 해당 퀘스트가 현재 설계한 구조로 구현 가능한지 테스트해 본 뒤 Class 속성의 추가, 삭제, 수정 과정을 통해 설계 완성

퀘스트 스크립트

퀘스트 Class를 Lua 스크립트를 사용해 입력 값과 함수(명령어)의 조합으로 구성할 수 있다

* 예를 들어 퀘스트 Class에 다음과 같은 기능을 추가할 수 있다
* 값의 크기 비교와 and, or 등의 연산자를 통해 더욱 복잡한 조건 설정 가능
* 작용 리스트의 순차적인 적용(예: 퀘스트를 진행하면 경비병이 와서 말을 걸고 카메라가 회전에 왕궁을 비춰준다. 카메라워크가 끝나면 자동으로 왕궁 안으로 맵 이동이 된다)
* 선택적인 요소의 유동적인 구현

퀘스트 관리

복잡하고 확장성 높은 퀘스트 구조는 필연적으로 버그를 양산한다

퀘스트 관리

다음의 안전 장치를 통해 잠재적인 오류 방지

* 필요에 따라 **한 번에 수행 가능한 퀘스트의 수**를 제한
* 장착 아이템 등을 보상으로 주는 퀘스트의 경우 유저의 인벤토리에 빈 칸이 있는지 무조건 체크하도록 추가
* 퀘스트 관련 아이템은 **판매, 투기, 파괴, 양도가 불가능**해야 하며 일반 인벤토리와 별도의 퀘스트 인벤토리를 만들어 놓음
* 효과적인 퀘스트 수행을 방해하는 MMO의 여러 문제를 해결하기 위해 **인스턴스 던전** 활용

퀘스트 설계 관리

유저 입장에서는 하나라고 생각되는 퀘스트도 구조적으로는 2개 이상인 경우가 상당히 많다

그래서 퀘스트 설계는 최소 다음 2개의 문서가 작업 되어야 한다

* 퀘스트 플레이 시나리오: 유저 입장에서 수행한 퀘스트에 대한 설계 문서(퀘스트 기본 실습을 통해 연습한 문서)
* 퀘스트 Class 문서: 실제 퀘스트 Table의 각 필드에 값이 입력된 형태의 문서
* 최소 단위의 퀘스트 + 퀘스트 묶음을 별도의 클래스로 지정해 관리하기도 한다

퀘스트에는 대표적인 꼼수가 많이 필요한 게임 디자인 분야 중 하나

구조적인 제약 사항 내에서 최대한 많은 것을 표현할수록 좋은 퀘스트가 된다

발전된 퀘스트 구현

콘텐츠 기획의 두 정점은 퀘스트와 레벨 디자인이라 할 수 있다.

**퀘스트** + **레벨 디자인** + 다양한 **선택** 대화 + **스크립트**를 이용한 컷 신 = **더욱 강력한 퀘스트의 구현**

타 콘텐츠와의 연계

타 콘텐츠 디자인과의 연계를 통해 더욱 강력한 몰입감을 줄 수 있는 퀘스트를 제작할 수 있다.

기존 구성된 각종 레벨 디자인, 아이템, 몬스터, NPC 등을 활용하거나, 해당 퀘스트를 위해 콘텐츠를 일부 수정, 추가할 수 있다.

예시: 숲 속의 잠자는 공주

기본 목적: 숲 속의 잠자는 공주를 깨우기 위해 공주의 꿈 속으로 들어가 악몽의 근원을 찾아 제거하는 퀘스트

세부 진행

* 공주를 클릭하면 ‘악몽의 펜던트’라는 아이템이 필요하다고 함
* 악몽의 펜던트를 착용한 채 공주를 클릭하면 ‘악몽’이라는 이름의 던전에 입장. 해당 던전의 구조는 다름 아닌 왕궁 맵이고 몬스터는 왕궁을 침공한 각종 악마들
* 유저는 던전을 깨면서 공주가 어린 시절 목격한 충격적 사건(악마가 왕을 죽이고 왕으로 변신)을 보게 됨
* 던전을 클리어하면 공주가 깨어나고 ‘왕을 죽여라’라는 새로운 퀘스트 받게 됨

레벨 디자인과 연계

다음 요소들을 적절히 활용, 퀘스트를 강화할 수 있다

* 맵의 분위기
* 동선
* 각종 게임 오브젝트 배치
* 각종 맵 트리거 배치
* 4번 항목을 활용한 각종 퍼즐 요소(잠긴 문, 열쇠, 엘리베이터 등)

대화 Class 강화

다양한 형태의 대화(선행 조건 체크, 선택지, 대화 결과에 따른 보상) 등은 2D 프로젝트를 통해 실습한 것처럼 스크립트를 통해 구현할 수도 있으나 Class화 시켜 놓는다면 더욱 효과적인 관리 가능

퀘스트 클래스에서의 다양한 퀘스트 **수행 조건** 속성들을 대화 클래스에 응용, 적용 가능

**선택지**를 통해 자연스런 대화 주제의 선택으로 퀘스트를 받거나 다양한 형태의 분위기를 선택할 수 있다.

대화 대상, 대화 결과 등에 따라 **진행 루트**나 **보상 제공**

스크립트를 이용한 이벤트 컷 신

게임의 리소스를 활용, 스크립트, 캐릭터 배경, 카메라를 제어해 다양한 이벤트 컷 신 제작

좋은 퀘스트 만들기

퀘스트 수의 적절한 분배

레벨 별, 클래스 별, 지역 별 적절한 분배

* 퀘스트 내용에 대한 파악도 되지 않은 채 수십 개의 퀘스트를 쓸어 담거나
* 퀘스트 없이 일반적인 게임 플레이만 오랜 시간 반복하게 되는 구간 체크

재도전 난이도 조절

10대 마왕을 죽여야 보상을 받는 퀘스트

* 9번째 마왕을 잡다 사망 🡪 처음부터 다시해라 🡪 ‘나는 누구? 어디에서 왔지? 나라와 민족의 미래는 어떻게 될 것인가? 세계 평화를 위한 길은 무엇인가? 🡪 게임 접자

적절히 설계하면 게임의 몰입도를 높일 수 있지만 잘 못 쓰면 치명적 이탈 요소가 될 수 있다.

분기의 적절한 구성

퀘스트를 깼더니 산신령이 금도끼, 은도끼, 쇠도끼 중 보상을 고르라 한다

* 금도끼: 욕심쟁이라며 경험치만 주고 사라짐
* 은도끼: 은도끼만 주고 사라짐
* 쇠도끼: 금, 은, 쇠도끼 다 주고 사라짐

위의 퀘스트가 쓰레기인 이유

* 선택의 결과를 예측할 수 없다.
* 보상의 편차가 크다: 개발자 입장에서 균형을 맞췄다고 생각한 요소도 유저 입장에서는 플레이 패턴, 선호도 등의 차이에 따라 큰 편차가 발생할 수 있다.
* 분기의 의미가 없다.

A분기를 가지고 어떤 이득을 얻을 수 있고 어떤 플레이를 할 수 있을 까에 대한 고민

싫은 일을 강제로 시키지 않기

싫은 일을 시키는 퀘스트는 보상이 커도 유저들의 욕을 먹는다

대표적인 싫은 일

* 똥개 훈련 식 동선
* 봇 사냥과 다름없는 퀘스트 조건(예: 늑대 100마리 잡아와) 🡪 도전 과제와 퀘스트 설계의 주 목적 차이는?
* 도전 과제: 플레이를 조금 더 전문화 목적, 쉬운 거부터 점점 어려워진다
* 퀘스트: 플레이를 스토리에 맞게 가이드 해주는 역할
* 종잇장 HP에 선공 성향을 가진 NPC를 이웃 마을까지 호위하기
* 사막에서 바늘 찾기