Requirements 3

Software Engineering Team Project
SeoulTech

대전 모드 개발

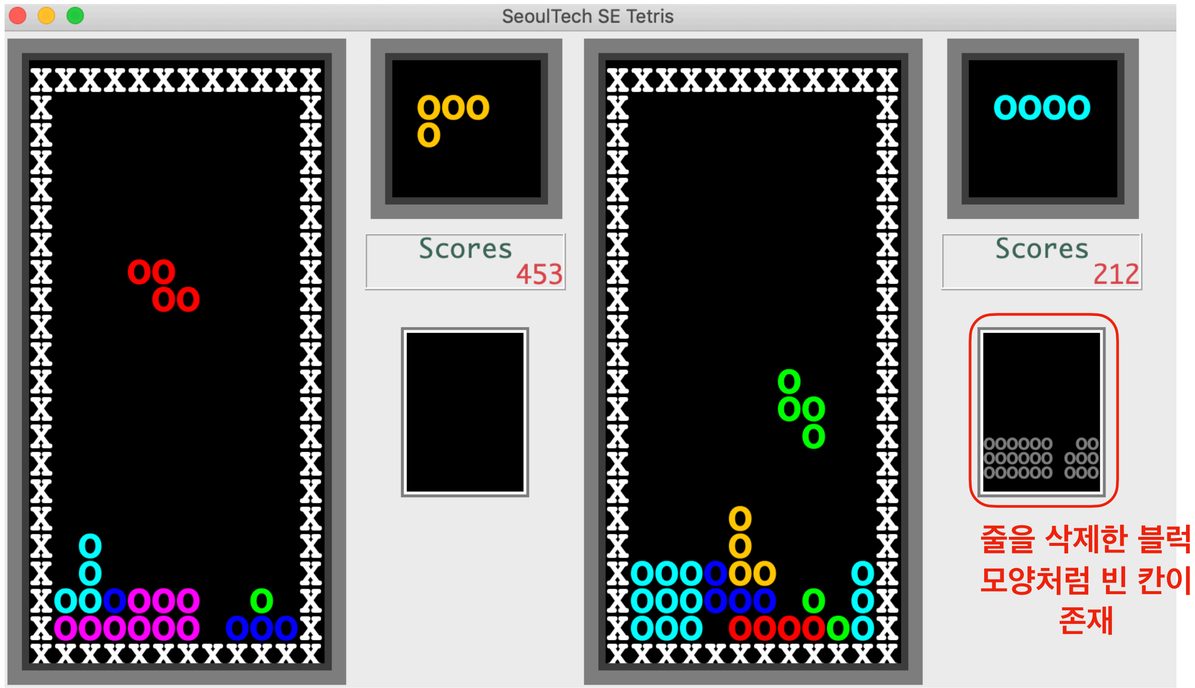
- 3차 요구사항의 핵심 내용은 게임에 대전 모드를 추가하는 것입니다.
  - 하나의 PC에서 2명이 대결하는 대전 모드 / 네트워크를 통해 대결하는 P2P 대전 모드.
- 직접적으로 제시되는 요구사항 외에도, 이를 적절히 구현하고 높은 품질의 게임을 만들기 위해 같이 수정되어야 할 부분은 직접 파악하여 처리해야 합니다.
- 현재의 프로그램 설계가 대전 모드를 추가하기에 적절하지 않다면 리팩토링을 진행합니다.
  - 이미 모듈화가 잘 되어있으면 매우 쉽게 대전 모드 개발을 완료할 수 있습니다.
- 명확히 언급되지 않은 부분에 대해서는 요구사항과 어긋나지 않는 한 자유롭게 구현하여도 좋습니다.
  - e.g.) 게임 화면 구성, UI설계, 추가적인 게임 정보 표시 등.

대전 모드 기본 기능

- 대전 모드는 시작 메뉴에서 선택하여 진행할 수 있습니다.
- 처음 게임 시작시 일반, 아이템, 시간제한의 3가지 모드 중 하나를 선택합니다.
- 일반 모드는 2명의 플레이어가 각자 테트리스를 진행하고, 각 플레이어가 삭제한 줄이 다른 플레이어의 보드에 추가되어 공격하는 방식입니다.
- 아이템 모드는 대전 모드 진행시 아이템이 등장하는 것 외에 일반 모드와 동일합니다.
- 시간제한 모드는 더 이상 블럭을 쌓을 수 없게되면 게임이 끝나는 조건 외에,
    - 타이머를 정해서 일정 시간이 다 지났다면 그 시점에 점수가 더 높은 사람이 승리하는 방식입니다.

대전 모드 게임 방식

- 대전 모드에서는 한 플레이어가 2줄 이상 블럭을 삭제하면,
  - 삭제된 줄들이 가장 마지막에 들어온 블럭 부분을 - 그 줄을 삭제되게 만든 블럭 - 제외하고 상대 플레이어에게 넘어갑니다.
  - 넘어간 줄들은 상대방의 넘어간 블럭 표시부분에 나타납니다.
  - 이 줄들은 플레이어가 현재 블럭을 처리하고 나면, 다음 블럭이 생성되기 전에 보드에 추가됩니다.
- 오른쪽 예제에서는 L자형 블럭으로 3줄을 삭제하여 상대방에게 줄이 넘어간 모습을 보여줍니다.



대전 모드 게임 방식

- 상대방에게 넘어간 블럭은 상대방 보드의 아래쪽으로 넘어간 형태 그대로 추가됩니다.
- 직접 쌓은 블럭과 구분하기 위해 추가된 블럭은 회색으로 처리합니다.
    - 색상은 자유롭게 결정해도 무방합니다.
- 기본 게임과 마찬가지로 플레이어 중 어느 한 쪽이 더 이상 블럭을 쌓을 수 없게 되면 게임이 종료됩니다.
- 대전 모드에서는 한 플레이어가 더 이상 블럭을 쌓지 못하는 상태가 되면, 다른 플레이어가 승자가 됩니다.
- 누가 승자인지를 화면에 표시해 주고, 그 이후는 일반적인 게임 종료와 동일하게 처리합니다.
    - 단, 대전 모드에서는 스코어 보드로 기록을 관리하지 않습니다.
- 그 밖에 아이템 사용에 관한 규칙이나 다른 여러 규칙은 자유롭게 결정해도 좋습니다.

대전 모드 공격관련 추가내용

- 상대 플레이어에게 넘길 수 있는 최대 줄 수는 10줄입니다.
- 여러 번의 공격으로 추가되는 줄은 현재 넘어간 줄의 아래쪽으로 추가됩니다.
- 추가될 줄이 표시되는 부분에 이미 10줄이 차 있는 경우, 줄을 삭제해도 상대방에게 넘어가지 않고 무시됩니다.
- 현재 10줄이 다 차있지 않더라도 넘어온 줄의 수를 더하여 10줄이 넘게 된다면, 제일 아래쪽 부분을 잘라내고 추가합니다.

대전 모드 개발 Tip

- Multi-thread 프로그래밍 연습을 하고 싶다면, 각각의 플레이어 화면을 두 개의 독립된 윈도우로 실행하도록 설계해도 좋습니다.

- 하나의 윈도우에서 Board를 두 개 만든다면 키 입력을 이벤트로 받아 처리하는 부분을 전체 윈도우에 추가한 뒤, 각각의 Board를 따로 처리해야 합니다.

  - 일반적으로 event가 focus가 있는 객체 - 현재 활성화 되었다고 판단되는 객체 - 에만 전달되기 때문입니다.

P2P 대전 모드 개발

- Peer-to-Peer (P2P) 대전 모드는 시작 메뉴에서 P2P 대전 모드를 골라 시작합니다.
    - 일반 대전 모드와 모든 것이 동일하지만, "2개의 PC"에서 테트리스 게임을 네트워크로 연결하여 대전을 진행합니다.
- P2P 대전 모드에서는 서버 또는 클라이언트를 고를 수 있고, 클라이언트가 서버에 접속하여 게임을 진행합니다.
- 서버를 선택한 경우, 클라이언트가 접속할 수 있도록 IP주소를 화면에 표시하고 대기합니다.
- 클라이언트에서는 IP주소를 입력하여 서버에 접속합니다.
- 연결이 완료되면 서버/클라이언트가 이를 알려주고, 대기 상태로 기다립니다.
    - 이 때 서버에 해당하는 플레이어가 일반, 아이템, 시간제한 모드를 선택할 수 있습니다.
- 서버와 클라이언트가 모두 '게임 시작' 버튼을 누르면 대전을 시작합니다.

P2P 대전 모드 게임 방식

- 대전 모드와 마찬가지로 내가 플레이 하는 화면과 상대방이 플레이 하는 화면이 모두 보여야 합니다.
- 각 플레이어의 조작에 따라 블럭 등은 실시간으로 양측의 화면에 동시에 표시되어야 합니다.
    - 동일 로컬 네트워크 상에서 연결되었을 때 키 입력과 화면 표시 지연 200ms 이하.
- 게임이 완료되면 대전 모드와 동일하게 승패를 표시하고, 네트워크 연결 직후의 대기상태로 돌아갑니다.
- 이 상태에서 다시 '게임 시작' 버튼을 눌러 새로운 게임을 시작할 수 있습니다.

P2P 대전 모드 추가 내용

- P2P 대전 모드에서는 네트워크를 이용하므로, **네트워크 문제가 발생하면 이에 대해 처리**해 주어야 합니다.
  - 단순히 네트워크가 느려 전송이 지연되는 경우 - 일명 "랙 걸린 상태" - 이를 플레이어에게 표시해 줍니다.
  - 네트워크 연결 자체가 일정 시간 이상 끊어진다면 에러 메시지를 표시하고 P2P 대전 모드를 처음 들어갔을 때의 화면으로 돌아갑니다.
  - 전송 지연이나 연결 끊김의 기준은 게임 플레이가 원활하게 될 수 있도록 직접 정의하여 관리합니다.
- 편리한 접속을 위해 최근에 접속했던 IP를 저장하고 P2P 대전 모드에서 클라이언트가 이를 확인하여 입력할 수 있게 합니다.
- 그 밖에 네트워크 접속 과정이 잘 진행되는지 확인할 수 있도록 적절한 메시지를 플레이어에게 표시합니다.
- 네트워크 접속 과정이 잘 진행되는지 확인할 수 있는 **자동화 된 테스트를 작성**합니다.

P2P 대전 모드 개발 Tip

- 대전 모드와 동일하게 게임을 진행하므로 양쪽 Board 등에 표시할 정보만 네트워크를 통해 주고 받도록 하여 빠르게 구현할 수 있습니다.
- 단순한 정보 교환이면 충분하므로 기본적인 Socket 프로그래밍으로 구현 가능합니다.
- 정보 교환을 위해 교환이 필요한 객체의 인스턴스를 Serialization하여 네트워크로 전송하고, 이를 Deserialization하여 바로 사용할 수 있습니다.
  - 단, 조작과 표시 사이의 지연을 줄이기 위해 보낼 정보를 적절하게 구성할 필요가 있습니다.
- 기본적인 Java의 기능으로도 충분히 구현 가능하나, 보다 다양한 기능에 대한 지원을 바라는 경우 관련된 라이브러리를 찾아 사용해도 됩니다.

도전 과제

- 주어진 요구사항을 모두 개발하고 여유가 있는 팀은 Git에서 새롭게 branch를 만들고 다음의 기능을 개발해 보세요.
- **컴퓨터 대전 모드**: 친구가 없는 플레이어를 위해 대전 모드의 상대방을 컴퓨터가 플레이하도록 만들어 보세요.
  - 사용자의 키보드 입력을 알고리즘 등을 구현해 대신하는 방식으로 구현할 수 있습니다.
  - 컴퓨터 플레이어를 구현한 함수는 현재 보드 상태를 입력으로 받고 다음에 눌러야 할 키를 반환하고, 이런 함수를 0.5~1초 간격으로 호출하는 식으로 사람과 유사하게 동작하도록 합니다.
- **P2P 대전 모드 채팅 기능**: 연결 완료 후 대기 상태에서 간단한 메시지를 주고 받을 수 있게 해보세요.
- **다인 대전 모드**: P2P 연결 시 1:1이 아니라 서버에 여러 클라이언트가 접속해 대전을 할 수 있게 확장해보세요.
- 개발이 완료되어도 최종 평가를 위해 main branch에는 merge하지 않습니다.
- 최종 평가에서 도전 과제 부분은 따로 점수를 부여하지 않으나, 시연에서 많은 득표를 할 가능성이 있습니다.

1,2차 요구사항관련 업데이트

- 최종 평가에서는 1,2차 요구사항 중에 비기능적 요구사항에 대한 평가가 강화될 예정입니다.
  - Unit Test 등으로 코드가 적절히 잘 검증되도록 하세요.
    - 기말 평가에서는 Line Coverage를 70% 충족해야 합니다.
  - 코드 리뷰, 리팩토링, 버전관리 등이 잘 이루어졌는지도 확인합니다.
- 최종 평가에서는 2차 요구사항의 팀별로 설계/구현한 아이템에 대해 직접 정리한 요구사항 명세서 또한 적절히 작성되었는지 평가될 예정입니다.
  - 중간 평가에서 제시된 요구사항 명세 및 요구사항 관리에 대해 배운 내용을 참조하여 적절히 요구사항을 작성하시기 바랍니다.

주의 사항

- 이 문서는 대략적인 요구사항을 설명하고 있습니다.
  - 추가적, 세부적 요구사항에 대해서는 팀별로 주어진 요구사항을 분석하여 결정해야 하며,
  - 모호한 부분이나 팀 내에서만 결정해서 문제가 될 것 같은 부분은 e-class에 문의하여 정해야 합니다.
- 각 슬라이드의 요구사항은 기능적/비기능적 요구사항이 섞여서 제시될 수 있습니다.
- 명시적으로 작성되어 있지 않으나, 주어진 요구사항을 충족시키기 위해 암시적으로 요구되는 사항 또한 평가 시 확인할 수 있습니다.
- 최종 평가에서는 1, 2, 3차의 요구사항을 모두 평가합니다.
  - 3차 요구사항 개발을 완료하면 다시 한 번 지난 요구사항들을 점검해 보세요.
  - 실제 평가시 네트워크 속도가 많이 저하될 수 있으니 이를 고려하여 테스트를 준비하시기 바랍니다.