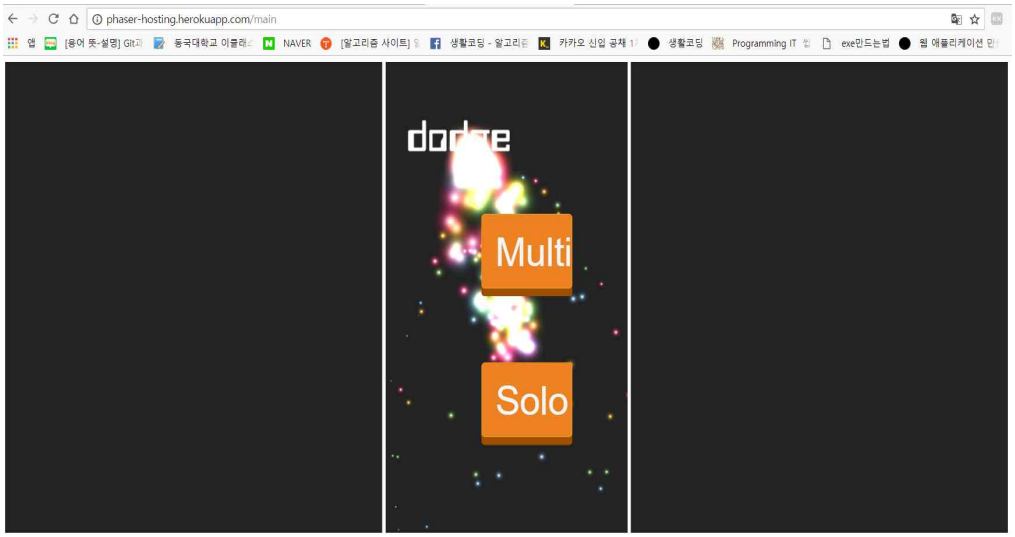
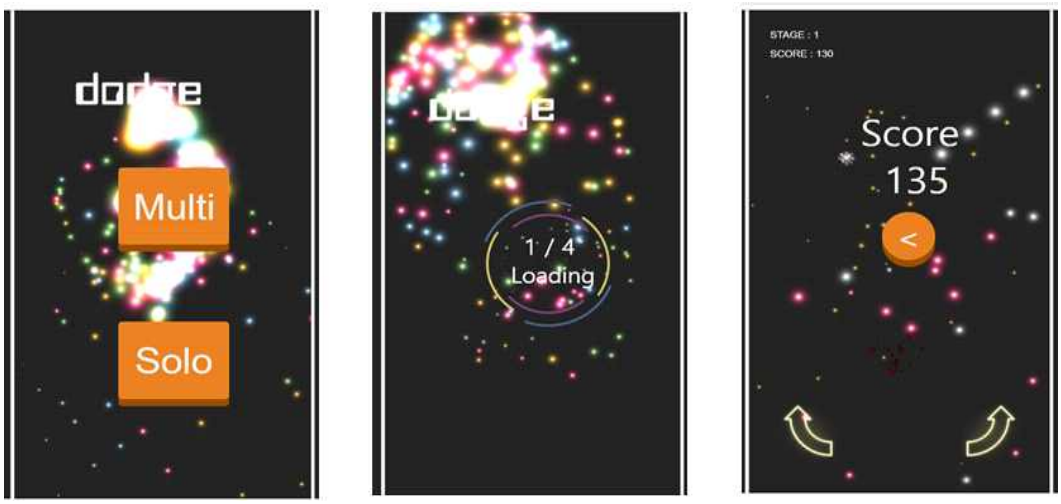
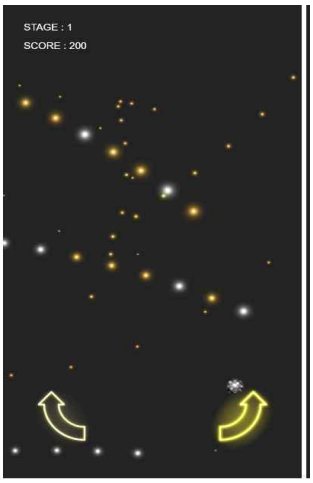
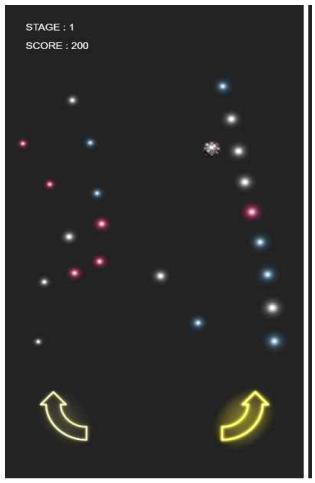
공개 소프트웨어 프로젝트 최종보고서

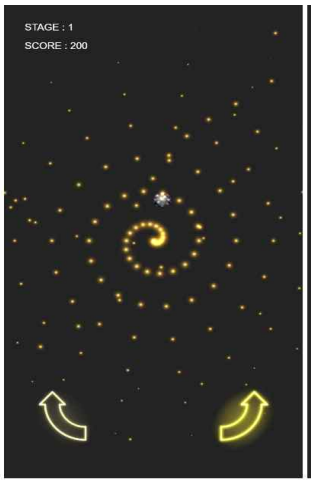
과제 수행원 현황						
수행 학기	2018년 3월~6월 1학기					
프로젝트명	Phaser 3 와 Node.js 를 이용한 멀티플레이 웹 게임 개발					
팀명	4조 Roly Poly					
	학과	학번	성명	성별	연락처	E-mail
팀장	컴퓨터공학과	2014112049	김인제	남	010 7769 5756	helios789@naver.com
팀원	컴퓨터공학과	2014112048	김형우	남	010 5742 4538	hwjw9599@naver.com
	컴퓨터공학과	2014112067	장현석	남	010 9260 3274	gustjr1259@naver.com
지도교수	교과목명	공개 sw 프로젝트				
	소속	컴퓨터공학전공				
	성명	손 윤 식 교수님				

# Key Words	웹 게임	멀티플레이	Phaser 3	멀티 플랫폼	튜토리얼
1.개발동기/목적/필요성 및 개발 목표	* 개발동기 및 목적: 기존 오픈소스 프로젝트들을 살펴보던 중에 주변에서 다가오는 적을 피하는 간단한 Dodge 게임을 발견했고 흥미가 생겨서 직접 게임을 플레이해보았다. 하지만 실제로 게임을 플레이 해본 결과 몇 가지 한계점을 찾을 수 있었다. 우선 게임을 플레이하기 위해 실행파일을 다운로드 받아야 하며 PC 에서만 플레이가 가능하며 유저간의 멀티플레이를 위해 서로의 IP 주소를 사전에 알고 있어야 한다는 점 이였다. 이러한 기존 프로젝트의 한계점을 개선하여 PC, 태블릿, 스마트 폰 등 어떠한 기기에서도 구동되는 멀티플레이 게임으로 확장하고 싶었다. 따라서 이때 채택한 확장 방식이 바로 웹 게임 이였다. 웹 은 기기에 상관없이 접근이 가능하므로 웹브라우저 상에서 구동되는 웹 게임으로의 확장을 개발 목표로 설정하였다. 웹 게임으로 확장하였을 때 의 장점은 우선 게임의 다운로드 없이 접속만으로 게임 플레이가 가능하며 링크 하나 만으로도 쉽게 게임에 접근할 수 있다는 점 등 이 있었으며 이러한 장점들이 기존의 프로젝트를 한 층 더 개선할 수 있다고 생각하여 Phaser 3 와 Node.js 를 이용한 멀티플레이 웹 게임 개발 프로젝트를 진행하게 되었다. * 개발목표: 프로젝트의 최종 개발 목표는 접속만으로 멀티플레이가 가능한 웹 게임의 구현 및 배포 이다.				

	이때 웹 게임을 구현하기 위해 2018년 2월 말에 출시된, 최신 오픈 소스인 Phaser 3 을 사용하였다. Phaser 3 은 웹 게임 개발을 위한 다양한 기능을 갖춘 HTML 게임 프레임 워크이다. 웹 게임의 배포를 위해선 Node.js 를 사용하여 서버를 구현하고 Heroku 라는 해외 Node.js 무료 호스팅 서비스를 이용하여 배포를 진행하였다. 멀티플레이 게임의 구현을 위해선 Node.js 의 모듈 중 하나인 Socket.io를 사용하여 실시간 유저, 서버간의 데이터 통신으로 멀티플레이를 구현하였다. 따라서 최종적인 프로젝트의 개발 형태는 하나의 웹 사이트 이며 접속만으로 즉시 게임 플레이가 가능하게 구현하는 것 이 개발 목표였다.
2.최종 결과물 소개	* 최종결과물 소개 및 사진 필히 첨부  ▲ 최종적으로 배포 된 웹 사이트에 접속 시의 화면  ▲ 최종 결과물인 웹 게임의 메인, 로딩, 게임종료 화면 Phaser3, Node.js, Socket.io 를 통해 개발한 웹 게임을 무료 호스팅 서비스인 Heroku를 통해 배포하였다. [http://phaser-hosting.herokuapp.com/main] 로 접속하면 게임을 바로 플레이 해 볼 수 있다.







▲ 게임의 실제 플레이 화면

게임이 진행될수록 점수와 난이도가 상승하며 이에 따라 다양한 적 의 패턴을 구현하였으며 게임 종료 시 최종 점수와 메인화면으로 돌아가는 버튼을 구현하였다.

3.프로젝트 추진 내용

프로젝트 진행과정

* 해당항목은 추후 디렉터리 및 보도 자료로 활용 예정이오니 상세하고 쉽게 작성 요망

목 표	3월		4월				5월				6월	
	3	4	1	2	3	4	1	2	3	4	1	2
오픈소스 구조 분석	→	→	→									
Phaser3 관련 공부	→	→	→	→								
Node.js 공부			→	→	→	→						
웹 게임 구현					→	→	→	→				
서버 구현							→	→	→	→		
UI 구현 및 배포									→	→	→	
테스팅 및 튜토리얼 제작											→	→

Phaser 3, Node.js, Socket.io 등 프로젝트 진행에 필요한 기술들을 우선 공부한 후 프로젝트 개발을 진행하였다.

• **Phaser 3** 는 공식 API 문서 및 Phaser 3 커뮤니티에 등록되어있는 게임 개발 튜토리얼, 예제를 적극 활용하여 게임 개발에 필요한 기능들을 학습하였다.

• **Node.js** 와 **Socket.io**는 "모던 웹을 위한 Node.js 프로그래밍 3판 - 윤인성, 한빛미디어" 책을 구입하여 학습하였다.

• **웹 게임 구현** 은 Phaser 3 로 진행하였으며 **서버 구현** 은 Node.js 로, **멀티플레이 기능**은 Node.js 의 Socket.io 를 통해 유저, 서버간의 실시간 데이터 통신을 구현하였다.

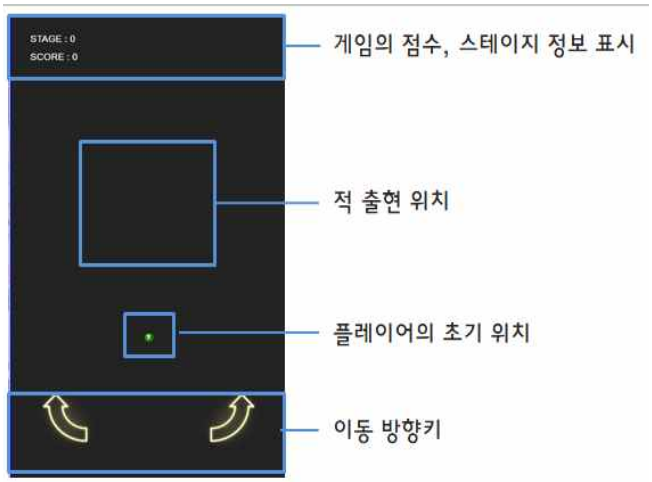
• **UI 구현** 은 HTML, CSS를 통해 구현하였으며 이를 웹 사이트의 형태로

배포하기 위해 무료 Node.js 호스팅 서비스인 Heroku 를 이용하여 배포하였다.

• **테스팅** 은 PC, 태블릿, 스마트 폰 에서 진행하였으며 해상도 문제 및 멀티플레이 기능을 테스트 하였다.

• 게임 개발이 완료 된 후 **게임 제작 튜토리얼** 또한 문서화 하여 배포하였다.

□ 게임 개발 과정

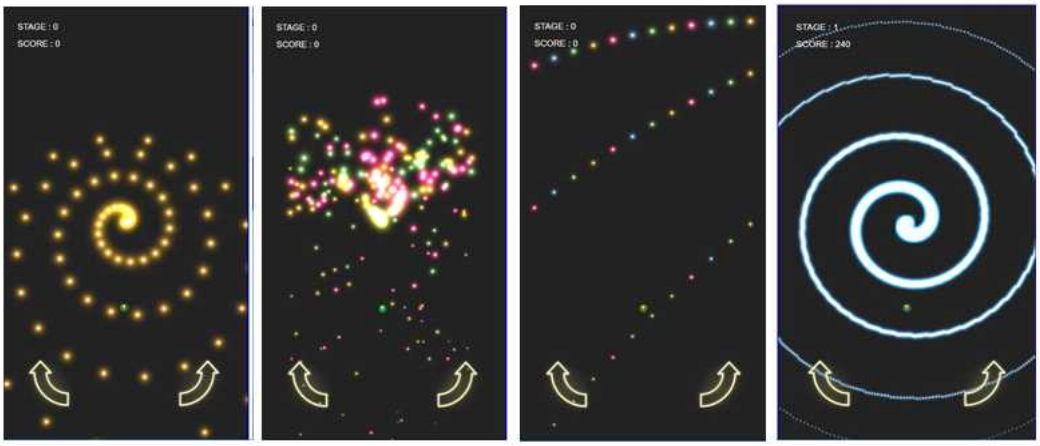


▲ 게임의 기초적인 디자인 구상

우선 Dodge 게임에서 필요한 기능 및 요소들의 배치와 같은 게임 디자인을 진행하였다. 게임의 점수와 스테이지 정보를 상단에, 적 및 플레이어를 중단에, 이동방향키 표시를 하단에 배치하였다.
게임의 점수, 스테이지 는 플레이어의 생존시간에 비례하여 증가하며 이에 따라 좀 더 다양하고 어려운 적 패턴이 출현하게 되므로 난이도 또한 증가하게 구현하였다.

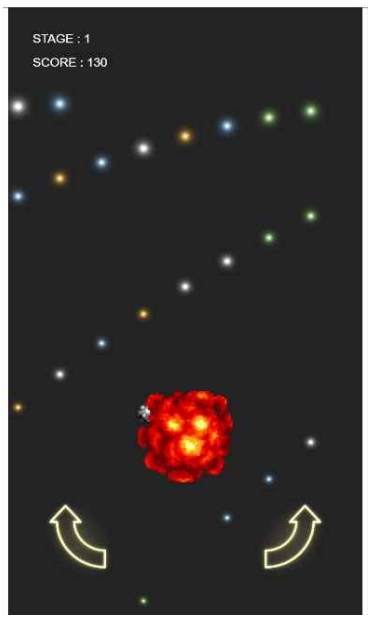
▲ 플레이어의 이동 구현

사용자의 입력에 따른 플레이어의 위치이동을 구현하였다.
Phaser 3에서 지원하고 있는 사용자 입력 기능을 통해 키보드 버튼 입력, 마우스 클릭, 스크린 터치를 모두 입력 받을 수 있게 구현하였다.
플레이어의 이동은 Phaser 3에서 지원하는 Sin, Cos 등의 수학연산을 이용하여 원 좌표 개념을 도입하여 플레이어가 원을 따라서 원형으로 이동할 수 있도록 구현하였다.



▲ 다양한 적 패턴 구현

게임의 완성도 및 난이도 상승을 위해 다양한 적 패턴을 구현하였다.
Phaser 3에서 지원하는 Particle 기능을 이용하여 다수의 적을 생성하고 이를 정해진 패턴대로 이동하게 하였다.
적과 플레이어의 충돌검사는 Phaser 3 의 자체 물리엔진을 이용하여 처리하였다.



▲ 플레이어 와 적 충돌 시 폭발하는 애니메이션 출력, 현재 점수 저장

□ 서버 개발 과정

이렇게 구현한 게임은 유저가 혼자서만 플레이 할 수 있으므로, 이를 멀티플레이가 가능한 게임으로 확장하기 위해 Node.js를 통해 서버를 구현하였으며 Socket.io를 통해 서버, 유저 간의 실시간 데이터 통신을 구현하여 멀티플레이 게임으로 확장하였다.

server.js

```
var express = require("express");
var path    = require("path");
var app     = express();
var server = require('http').Server(app);
var io = require('socket.io').listen(server);

app.use(express.static(path.join(__dirname + '/public')));

app.get('/main',function(req,res){
    res.sendFile("mainS.html", { root: path.join(__dirname, 'public') })
});
```

▲ Node.js를 이용하면 10줄도 안 되는 코드로 서버구현이 가능하다

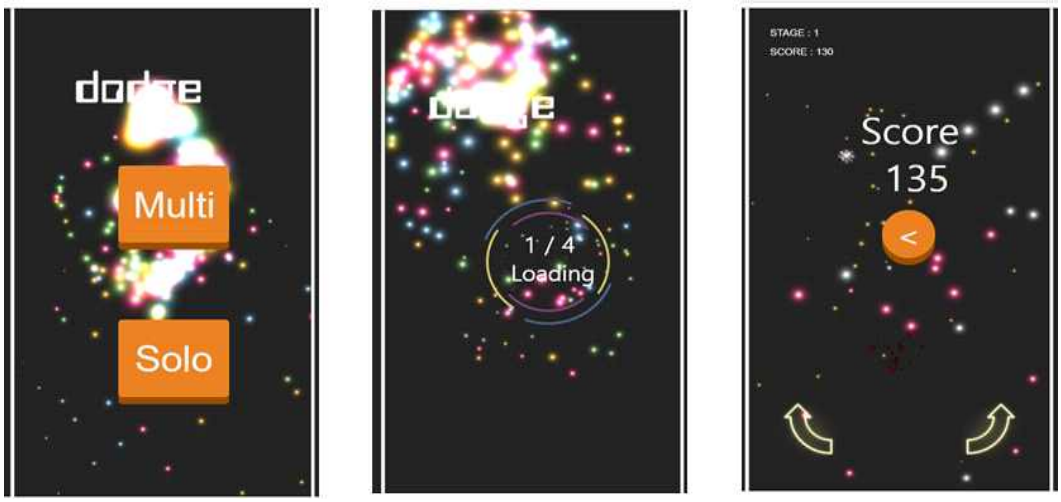
▲ 서버와 유저가 이벤트 발생을 기반으로 실시간 통신

멀티플레이 기능 구현을 위해 각 유저간의 위치 정보나 적과의 충돌 상태와 같은 정보들을 서버와 실시간으로 통신해야 한다.
이때 주기적으로 서버에서 모든 유저들의 위치정보를 받아오는 폴링 방식을 사용한다면, 서버에 매우 많은 데이터들이 실시간으로 전송되므로 단일 스레드를 기반으로 하는 Node.js 의 특성상 서버에 큰 과부하가 걸리게 된다. 이러한 방식은 실시간으로 정보 전송 및 업데이트가 진행되어야 하는 멀티플레이 게임에 적합하지 않다고 판단하였다. 따라서 실시간으로 빠르게 유저들의 정보를 서버에서 받아오고 다시 전송하기 위해 새로운 유저의 접속, 유저의 이동, 적과의 충돌, 유저의 접속 종료 등 과 같은 상황들을 하나의 이벤트 로 정의하고, 각각의 이벤트 발생 시 에만 서버에 이벤트 발생을 알리고, 서버는 나머지 유저들에게 다시 이 이벤트의 발생을 전송하는 이벤트 기반의 방식으로 서버를 구현하였다.

▲ 새로운 유저가 접속 이벤트 발생 시 유저들과 서버 사이의 통신 과정

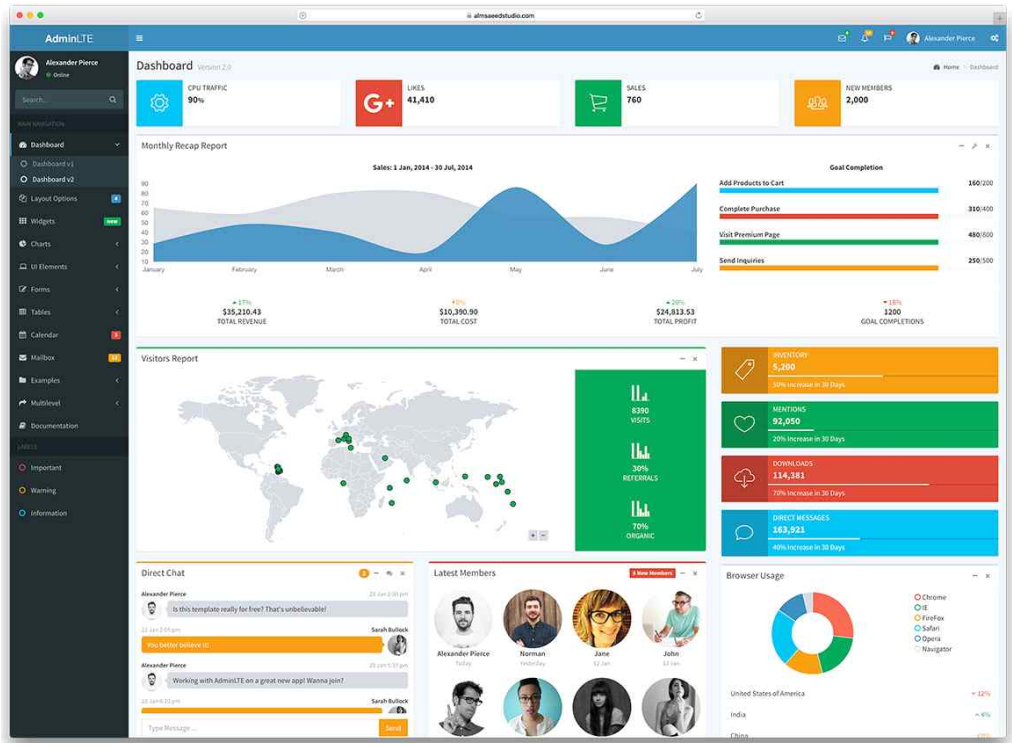
새로운 유저가 접속한 상황을 예로 들면, 위에 그림에서 알 수 있듯이 유저 측에서 새로운 유저 접속 이벤트를 발생시키고 서버는 이 이벤트를 받아 새로운 유저의 위치정보를 생성하여 나머지 다른 유저들에게 전송한다. 이처럼 이벤트 기반의 실시간 통신방식을 구현하여 서버의 과부하를 줄이고 실시간 멀티플레이 게임이 가능하게 하였다.

□ UI 개발 과정



▲ HTML, CSS 와 jQuery 를 이용하여 구현한 게임의 메인, 로딩, 종료 UI 화면

앞서 구현한 멀티 플레이 웹 게임의 UI를 구현하기 위해, Phaser 3에서 지원하는 UI 기능이 아닌 웹 자체의 기능인 HTML, CSS 와 라이브러리인 jQuery를 사용하였다.
Phaser 3 는 아직 출시 된지 4개월 밖에 되지 않았으며 오픈소스 이기 때문에 게임 개발에는 문제가 없지만 게임의 UI 구현 부분 에서는 디자인, 기능적으로 조금은 부족한 측면이 있었다. 따라서 Phaser 3에서 지원하는 UI 기능이 아닌, 웹 자체에서 지원하는 HTML, CSS 등을 사용하여 UI를 자체적으로 디자인하여 구현하였다.
게임 접속 시 의 솔로, 멀티 플레이 게임 모드 선택버튼, 메인화면, 멀티플레이 시 로딩 및 대기화면, 게임 종료 시 최종 점수 표시 및 메인화면으로 이동 버튼 등을 구현하여 게임이 원활하게 이루어 질 수 있도록 구현하였다.



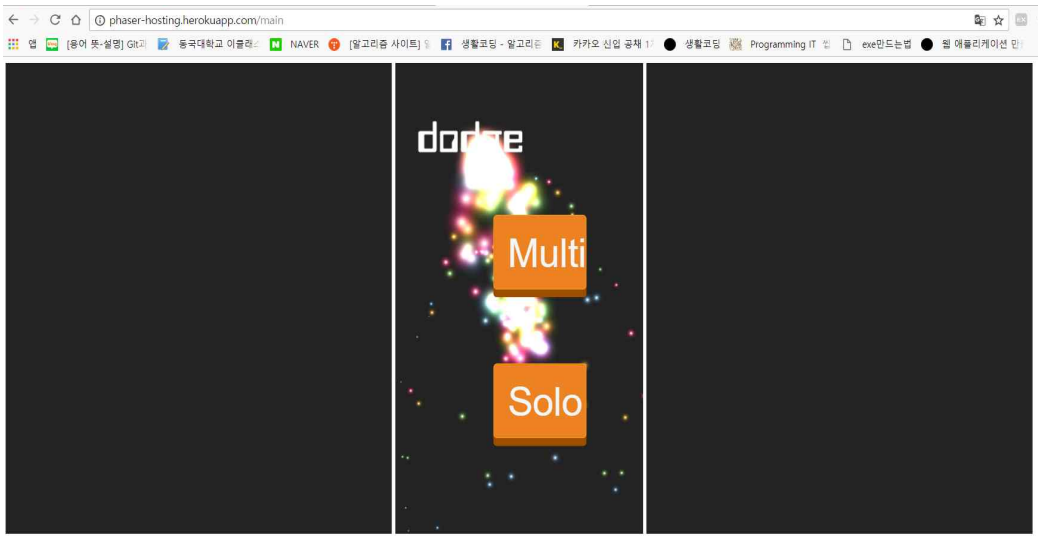
▲ 오픈소스 웹 UI 프레임워크 인 부트스트랩으로 구현한 웹 사이트의 예시

웹 게임에서 웹 자체의 CSS, javascript 등을 사용할 수 있다는 점은 매우 강력한 장점을 가진다. 바로 이미 구현되어 있는 수많은 오픈소스 웹 UI 프레임워크들을 바로 게임의 UI 로 적용할 수 있다는 것 이다. 부트스트랩, 마테리얼 UI처럼 이미 웹 디자인을 위한 엄청나게 많은 오픈소스 들이 존재하며 이들은 모두 매우 간편하게 사용할 수 있으며 PC, 모바일 해상도 모두를 지원하는 반응형 웹 디자인으로 구현되어 있다. 따라서 이러한 UI 프레임워크들을 도입한다면 게임 UI를 매우 빠르고 세련된 디자인으로 만들 수 있다는 장점이 있다.

□ 호스팅 과정

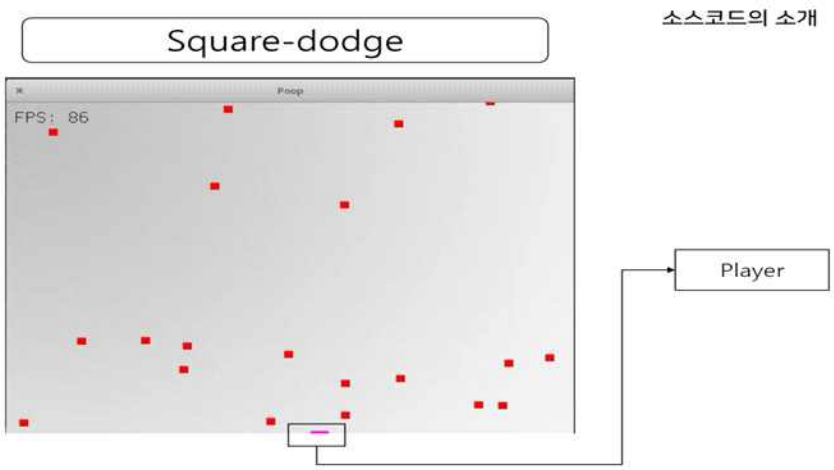
▲ Node.js 무료 호스팅 서비스인 Heroku를 이용하여 프로젝트를 배포하였다

최종적으로 구현된 게임을 웹 사이트 형태로 배포하기 위해 개인 서버를 구축하여 배포하는 것 보다는 호스팅 서비스를 이용하기로 결정했다. AWS와 같은 좋은 호스팅 서비스 들이 있지만, Node.js 호스팅을 지원하고 최적화가 되어있는 Heroku 라는 해외 무료 호스팅 서비스를 발견하여, 이를 통해 호스팅 및 웹 사이트 배포를 진행하였다. 또한 Heroku Github와 완벽하게 연동이 되어 기존에 프로젝트를 Github를 통해 버전관리 하고 있었는데, 이를 Heroku 에다 push만 하면 호스팅이 완료된다는 매우 편리한 장점이 있었다.



▲ 호스팅 이 완료된 후, 웹 사이트에 실제로 접속해 본 화면

□ 기존 프로젝트와의 차이점



▲ 기존의 프로젝트 실행화면

▲ 새롭게 확장한 프로젝트의 실행화면

기존의 프로젝트와의 차이점은,

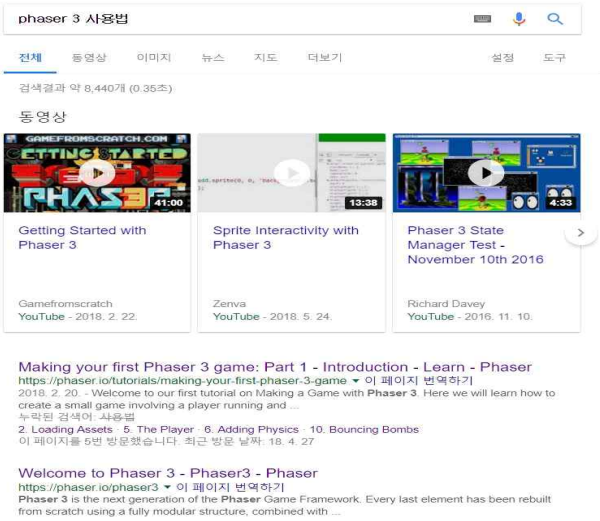
- 게임 플레이 환경이 PC에서 PC, 태블릿, 스마트 폰 등 **멀티 플랫폼**으로 확장
- 게임의 실행파일 다운로드 없이 **웹 사이트 접속**만으로 즉시 게임 플레이 가능
- 멀티 플레이 시 **서로의 IP**를 모르더라도 플레이 가능
- 디자인적 요소 추가로 인한 **게임의 완성도** 향상
- 게임의 메인, 로딩, 종료 화면 등 **UI 구현**
- 게임 제작 방법을 **튜토리얼** 로 문서화 하여 배포
- 추후 게임 기능 추가 등의 **업데이트 진행시** 게임 유저들의 추가적인 다운로드 없음
- Phaser 3, Node.js, Socket.io 모두 **javascript 언어**를 사용하므로 개발시간 단축 및 효율성 증가, 유지/보수 편리

등 이 있다.

4.기대효과	Phaser 3 와 Node.js를 이용한 멀티플레이 웹 게임 개발 프로젝트를 통해 얻을 수 있는 기대효과는 우선 Phaser 3 라는 오픈소스의 생태계에 기여할 수 있다는 점이다. Phaser 3 는 현재 가장 인기 있는 HTML5 게임 엔진 중 하나이며, 커뮤니티가 매우 활성화 되어 있으며 가장 최근에 릴리즈 되었고 따라서 매주 지속적인 업데이트와 변경사항이 있을 정도로 이제 막 세상에 나타난 오픈소스 이다. 이러한 Phaser 3을 이용하여 웹 게임을 개발하고, 나아가 Node.js, Socket.io 와 연동하여 멀티플레이가 가능한 게임으로 확장시킨 후 이 모든 과정을 튜토리얼 형식으로 문서화하여 배포하는 것을 통해 Phaser 3 의 오픈소스 생태계의 발전에 기여할 수 있다. 아직 Phaser 3 활용한 멀티 플레이 게임 개발 방법에 대한 자료가 거의 없기 때문이다. 또한 Phaser 3 에 대한 한국어로 된 자료가 현재 하나도 없는 실정 이므로, 이 프로젝트를 통해 국내에 웹 게임이라는, 아직은 다소 생소한 분야에서의 하나의 접근 방법을 제시할 수 도 있다. 아직 웹 게임 이라는 분야는 성장 중 이며 큰 잠재력을 가지고 있는 분야라고 생각한다. 불과 5년 전 까지만 해도 Adobe 사의 Flash 라는 기술을 활용한 일명 "플래시 게임" 이라는 게임이 웹 게임 시장을 독차지하고 있었다. 허나 보안등 여러 문제로 인해 Flash가 사용되지 않고 이를 대체하기 위해 HTML5 라는 표준안이 확장되었다. 따라서 HTML5을 이용한 지금의 웹 게임의 형태는 그 역사가 5년조차 되지 않은 것이다. 또한 게임 이라는 분야가 기존엔 화려한 그래픽 및 시각효과를 위해 성능이 좋은 PC 에서만 한정되었다면, 최근엔 출, 퇴근 및 쉬는 시간 등 자투리 시간에 간단하게 즐길 수 있고 다른 유저들과 쉽게 경쟁할 수 있는 모바일 게임으로의 확장이 진행되면서 게임 자체의 의미가 변화하고 있다. 웹 은 모든 기기에서 인터넷 연결만 되면 접속이 가능하다는 특성 때문에, 다운로드 없이 아주 쉽고 간편하게 즐길 수 있는 웹 게임의 성장 가능성은 앞으로도 무궁무진하다. 이미 국내에선 네이버 의 5분 게임 및 카카오의 게임별 서비스를 통해 웹 게임의 가능성을 확인할 수 있다. 특히 카카오의 게임별은 기존의 카카오톡 서비스에 개발된 웹 게임들을 바로 탑재한 형태로 웹 게임의 강력한 확장성을 알 수 있다. 또한 올해 5월, 카카오 게임즈 에서 제 1회 HTML 게임 공모전을 개최한 사실을 통해 웹 게임 시장이 성장하고 있으며 앞으로의 가능성 또한 높다는 것을 알 수 있다. 또한 웹 게임은 웹의 특성상 접근이 매우 쉽다는 장점이 있다. PC나 모바일 용 게임은 당연히 유저가 다운로드를 받아야 하지만, 웹 게임은 URL 링크 하나만 클릭하면 바로 즐길 수 있다는 점에서 SNS (소셜 네트워크 서비스) 상에서 매우 빠르게 홍보할 수 있고 많은 유저들을 끌어들일 수 있다는 장점이 있다. 이는 광고 수익모델을 적용할 수 있음을 뜻한다. Facebook 이나 Instagram 과 같은 SNS 에서만 봐도 현재 많은 사람들의 다양한 정보는 돈 과 직결된다. 여러 웹 게임을 모아놓은 웹 사이트에 많은 사람이 방문한다면 배너 광고 등을 통해 광고 수익도 충분히 올릴 수 있으며 나아가 커뮤니티 사이트 및 하나의 SNS로 발전 할 가능성 도 있다.

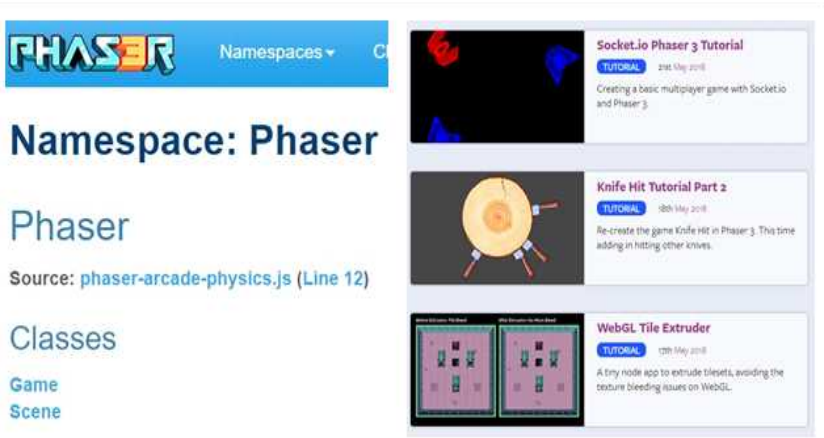
5.구현과정에서의 문제점 및 해결과정

□ Phaser 3 에 대한 자료 부족



▲ Phaser 3 에 대한 한국어로 된 자료 검색 결과 (0 건)

Phaser 3 의 사용법을 학습하는 과정에서 Phaser 3 가 올해 2월 말에 처음으로 릴리즈 된 비교적 최신의 오픈소스 이기 때문에, 한국어로 된 자료가 하나도 없었으며, 영어로 된 자료조차 많지 않으며 자료의 질이나 정확도 또한 떨어진다는 문제가 있었다. 이를 해결하기 위해 Phaser 3 의 공식 API 문서와 Phaser 3 자체 커뮤니티에 업로드 되어 있는 다른 개발자들의 게임 제작 튜토리얼 들을 적극적으로 활용하여 개발을 진행하였다. Phaser 3 커뮤니티가 매우 활성화 되어 있어 게임 개발 방법 및 기능에 대한 사용법 등 다양한 자료들을 얻을 수 있었다.



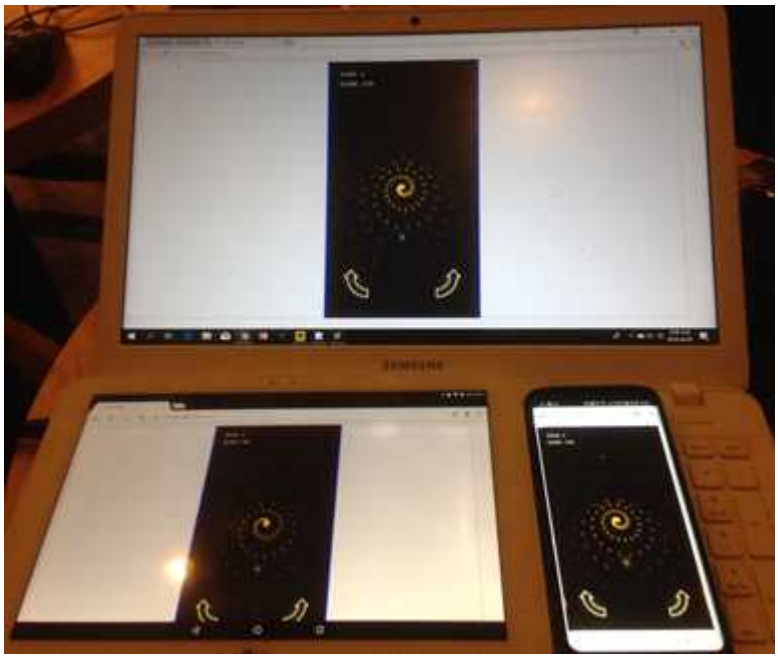
▲ Phaser 3 의 공식 API 문서와 커뮤니티에 등록된 다양한 개발 튜토리얼

□ 다양한 기기에서의 입력 문제

기존의 게임은 PC에서만 구동되므로 키보드 입력만 받으면 되었지만, 확장된 게임은 PC, 태블릿, 스마트 폰 의 키보드 버튼 입력, 마우스 클릭 과 스크린 터치 모두를 입력 받을 수 있어야 하므로, Phaser 3에서 지원하는 Input 기능을 활용하여 키보드, 마우스, 스크린 터치를 모두 지원하여 정상적으로 게임에서 입력받아 처리 할 수 있게 해결하였다.

□ 다양한 기기에서의 해상도 문제

개발한 웹 게임을 PC, 태블릿, 스마트 폰 등 다양한 기기에서 접속하여 플레이 하게 되며 이때 각 기기들 간의 해상도가 모두 다르고, 스마트 폰 사이에서도 기종에 따라 해상도가 차이 나므로 게임의 크기가 정해져 있지 않고 해상도에 따라 유동적으로 화면크기에 맞게 변화해야 한다는 문제점이 발생했다.
이 문제점을 해결하기 위해, javascript 코드를 통해 게임에 접속한 기기의 웹 브라우저 의 가로, 세로 크기를 통해 해상도에 맞게 게임 크기를 조정하는 방식을 채택하였다.



▲ 접속한 기기의 해상도와 화면 크기에 맞게 게임의 크기가 자동으로 조정

□ 게임 내에서 페이지 새로 고침 속도 문제

게임의 메인화면에서 솔로 및 멀티 게임 클릭 시, 초기에는 서버에서 지정해 둔 주소로 이동하여 페이지가 새로 고침 되면서 다시 로딩 되는 방식으로 구현하였는데, 성능이 안 좋은 스마트 폰이나 느린 와이 파이 연결 환경에서는 이 페이지 이동이 너무 느리게 진행되어 게임 플레이에 지장을 줬으며 또한 멀티 플레이 시 페이지 로딩이 느리면 게임이 동시에 시작이 안 된다는 문제가 발생하여, 이를 웹 페이지 안에서 다시 웹 페이지를 보여줄 수 있는 HTML 의 IFrame 태그를 활용하여 해결하였다. 게임 메인화면에 최초 접속 시 메인화면만 로딩 하는 것 이 아니라 보이지 않게 게임 자체도 로딩을 하므로 페이지의 이동 없이도 솔로, 멀티 게임을 바로 진행 할 수 있으며 페이지 로딩 시간 또한 매우 빠르게 개선되었다.

	□ 멀티플레이 게임 간의 동기화 문제 PC, 태블릿, 스마트 폰 사이의 성능 차이 로 인해 멀티 플레이 게임 접속 시 게임의 시작 시점이 다르다는 문제가 발생하였다. 이를 해결하기 위해 먼저 게임의 로딩이 끝난 유저가 다른 유저들의 로딩 완료를 기다리는 기능을 구현하였다. Phaser 3에서 지원하는 게임 정지 / 다시시작 기능을 이용하여 게임 로딩 완료시 우선 게임을 정지 한 후 다른 모든 유저가 접속 완료 되었을 때 동시에 게임을 다시 시작하여 멀티플레이 게임이 정상적으로 진행 될 수 있게 구현하였다. 또한 멀티플레이 게임 진행 중 에 각 플레이어들의 위치 정보와 같은 게임 데이터들을 주기적으로 통신하는 것 이 아닌 이벤트 방식에 기반 하여 통신하게 구현하였으며 서버와 유저 간에 데이터를 통신하는 과정도 최대한 간소화 하여 실시간 멀티플레이 게임이 가능하도록 구현하였다.
6.참고문헌	• 윤인성. (2016).「모던 웹을 위한 Node.js」프로그래밍 3판. 서울: 한빛미디어 • Phaser 3 API Document, "Phaser 3 API" , https://photonstorm.github.io/phaser3-docs/, (2018. 3) • Phaser.io, "Making your first Phaser 3 game", https://phaser.io/tutorials/making-your-first-phaser-3-game, (2018. 3) • Phaser 2 examples, "Phaser example", https://phaser.io/examples, (2018. 3) • Phaser 3 examples, "Phaser 3 example", https://labs.phaser.io/index.html, (2018. 3) • Emanulel Feronato, "Build a HTML5 game like "Knife Hit" with Phaser 3", http://www.emanueleferonato.com/2018/06/15/build-a-html5-game-like-knife-hit-with-phaser-3-using-only-tweens-and-trigonometry-adding-apples-and-slicing-them-too/, (2018. 4) • Emanulel Feronato, "The basics of responsive HTML5 games", http://www.emanueleferonato.com/2015/02/26/the-basics-of-responsive-html5-games/, (2018. 4) • ZENVA, "Phaser 3 multiplay game", https://gamedevacademy.org/create-a-basic-multiplayer-game-in-phaser-3-with-socket-io-part-1/, (2018, 4) • A MEAN Blog, "Node.JS & Socket.io 채팅사이트 만들기", https://www.a-mean-blog.com/ko/blog/%EB%8B%A8%ED%8E%B8%EA%B0%95%EC%A2%8C/_/Node-JS-Socket-io-%EC%B1%84%ED%8C%85%EC%82%AC%EC%9D%B4%ED%8A%B8-%EB%A7%8C%EB%93%A4%EA%B8%B0, (2018. 5) • CODEPEN, "CSS UI design", https://codepen.io/, (2018. 6)