



본 자료의 내용은 정기적으로 변경 또는 추가 될 수 있습니다. 비케이플래닛㈜는 사전 공지 없이 언제든지 기술 백서에 기재된 내용을 추가 또는 수정 할 수 있습니다.

미션 크리티컬 애플리케이션을 위한 OutSystems 로우 코드 플랫폼은 비주얼 모델 기반 개발, AI 기반 로우 코드 및 클라우드 네이티브 플랫폼을 사용하여 비즈니스에 중요한 엔터프라이즈 애플리케이션을 개발합니다.

2024 © BK Planet Co., Ltd.

본 자료는 비케이플래닛(주)의 사전 승인 없이 임의로 무단 복사 또는 복제하여 배포할 수 없습니다.

관련 사항 질문을 위한 연락처:

Tel. 02-516-1244 / bkplanet@bkplanet.com

Document Version 1.0

**목차**

[**1. 개요 3**](#_heading=h.30j0zll)

[1.1 시나리오 4](#_heading=h.1fob9te)

[**2. 빌드 방법 (How-to) 5**](#_heading=h.2et92p0)

[2.1 자원 (Resources) 5](#_heading=h.tyjcwt)

[2.2 시작하기 5](#_heading=h.3dy6vkm)

[2.2.1 비주얼 스튜디오 (Visual Studio) 설치 5](#_heading=h.lso1x5i8p9d)

[2.2.2 인티그레이션 스튜디오 (Integration Studio) 설치 6](#_heading=h.75cpqw6lnzsh)

[2.2.3 서비스 스튜디오 (Service Studio) 사용 7](#_heading=h.1t3h5sf)

[2.3 C# 코드 추가 9](#_heading=h.2s8eyo1)

[2.4 NuGet 패키지 추가 15](#_heading=h.17dp8vu)

[2.5 마무리하기 20](#_heading=h.3rdcrjn)

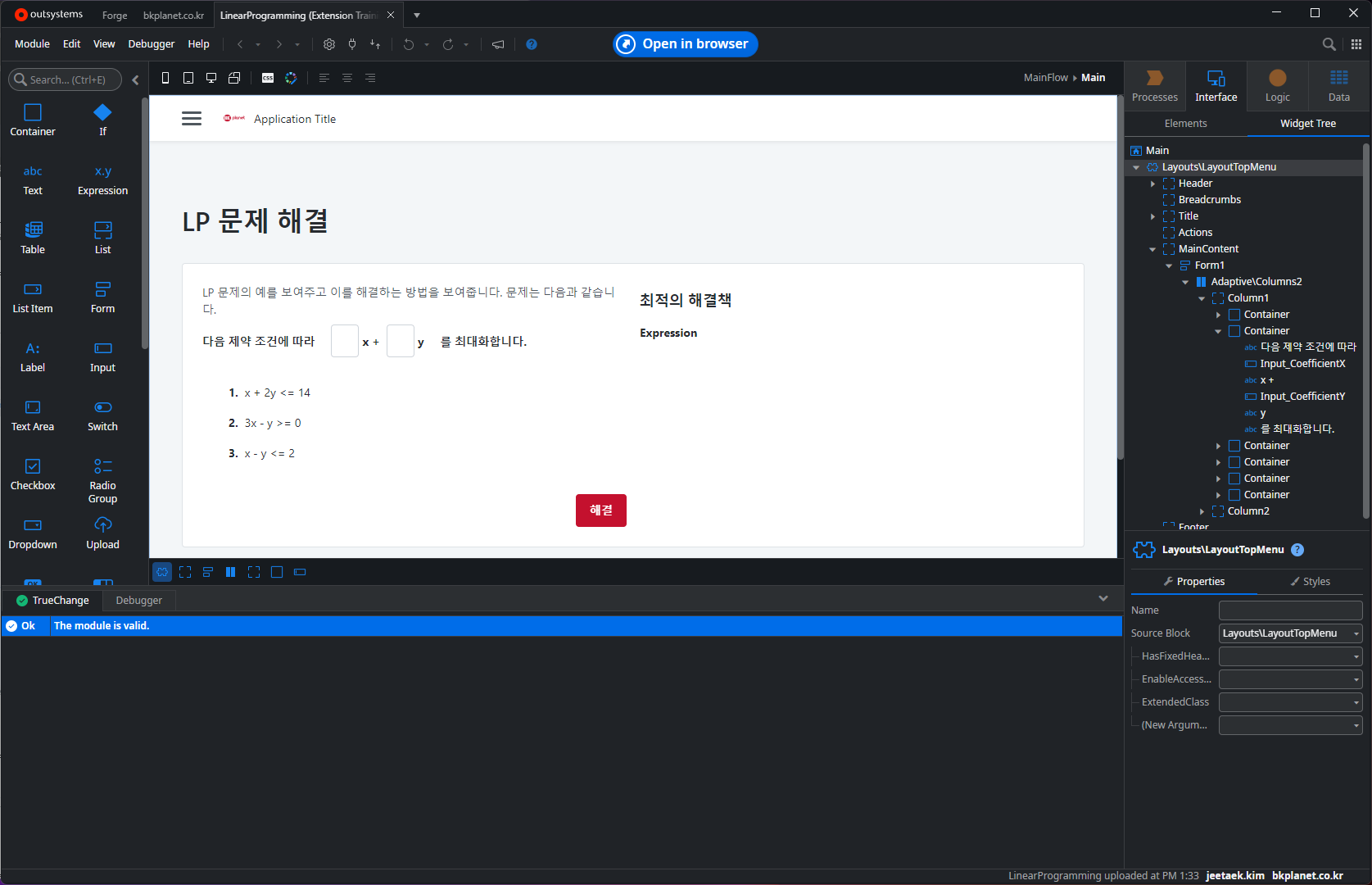
[**5. 참조 링크 23**](#_heading=h.26in1rg)

# 1. 개요

이 교육 매뉴얼에서는 OutSystems IDE 인 Service Studio 와 Integration Studio, Microsoft 의 Visual Studio 를 사용하여 Extension Module 을 활용해 C# 코드와 NuGet 패키지를 다루는 방법에 대해 학습합니다.

사전에 준비된 Extension Training 애플리케이션의 oap 파일을 활용하여, 간단한 실행 화면을 생성합니다.

최종적으로 브라우저에서 앱을 열고 사용해 볼 수 있습니다.



## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

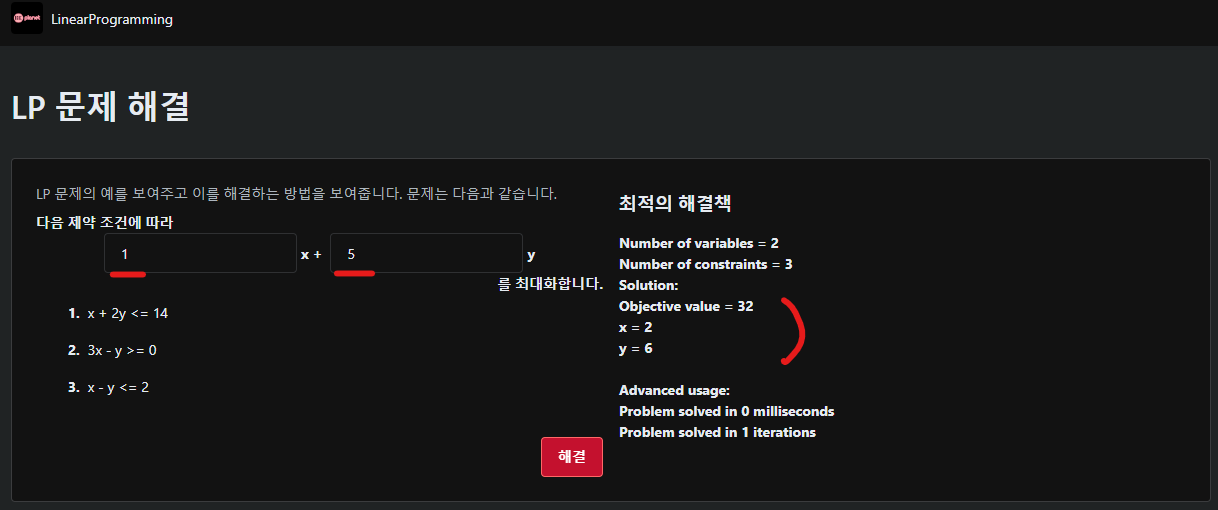
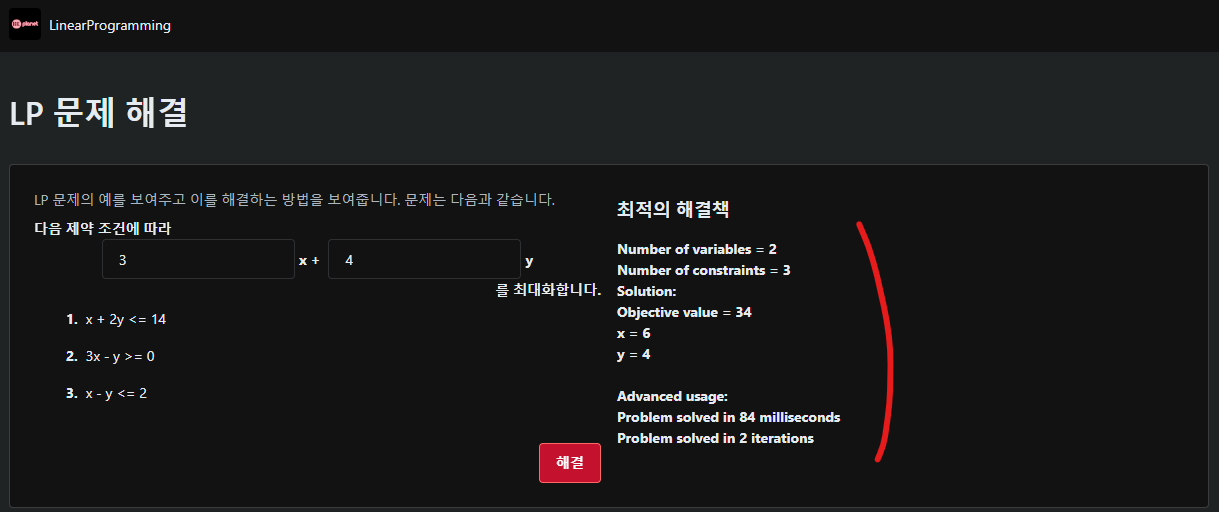
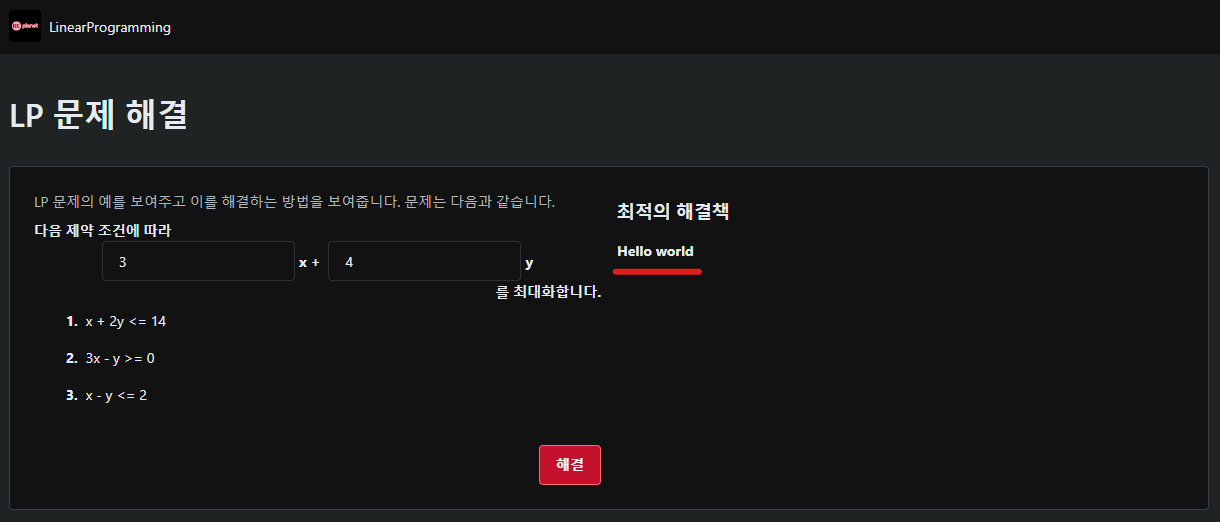
## 

## 1.1 시나리오

Google 의 OR-Tools 를 활용하여 선형 최적화 기능을 구축하고자 합니다. 이 화면에서는 C# 코드와 NuGet 패키지를 다루는 방법을 습득하고자 합니다. 이 교육 매뉴얼에서는 1개의 App과 1개의 Module, 1개의 Action 을 생성합니다.

* Extension Training: oap 파일로 제공되는 Reactive Web App
* LinearProgramming\_Eng: C# 코드가 입력되는 Extension Module
* MPSolver: 선형계획법 로직이 위치하는 Action

이 교육 매뉴얼은 최종적으로 3개의 결과를 브라우저로 출력합니다.



# 2. 빌드 방법 (How-to)

이제 시나리오를 알았으므로 이 가이드에 따라 LinearProgramming\_Eng 모듈을 빌드합니다.

## 2.1 자원 (Resources)

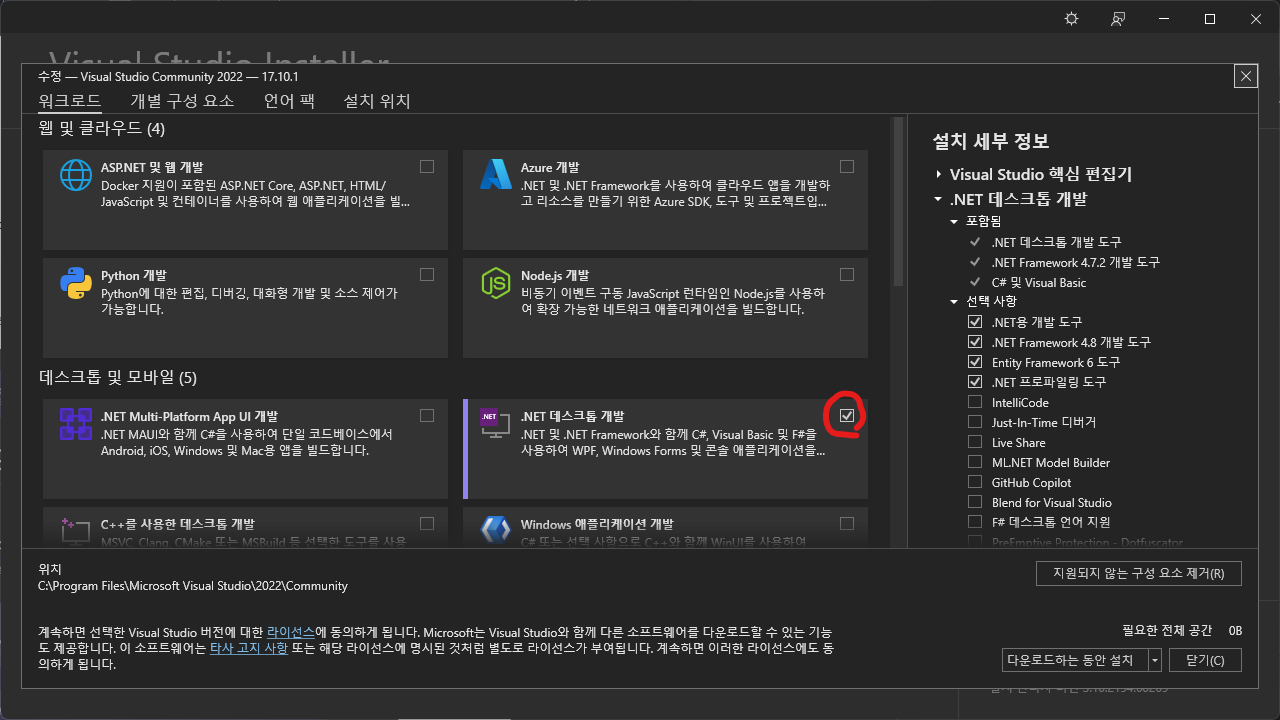
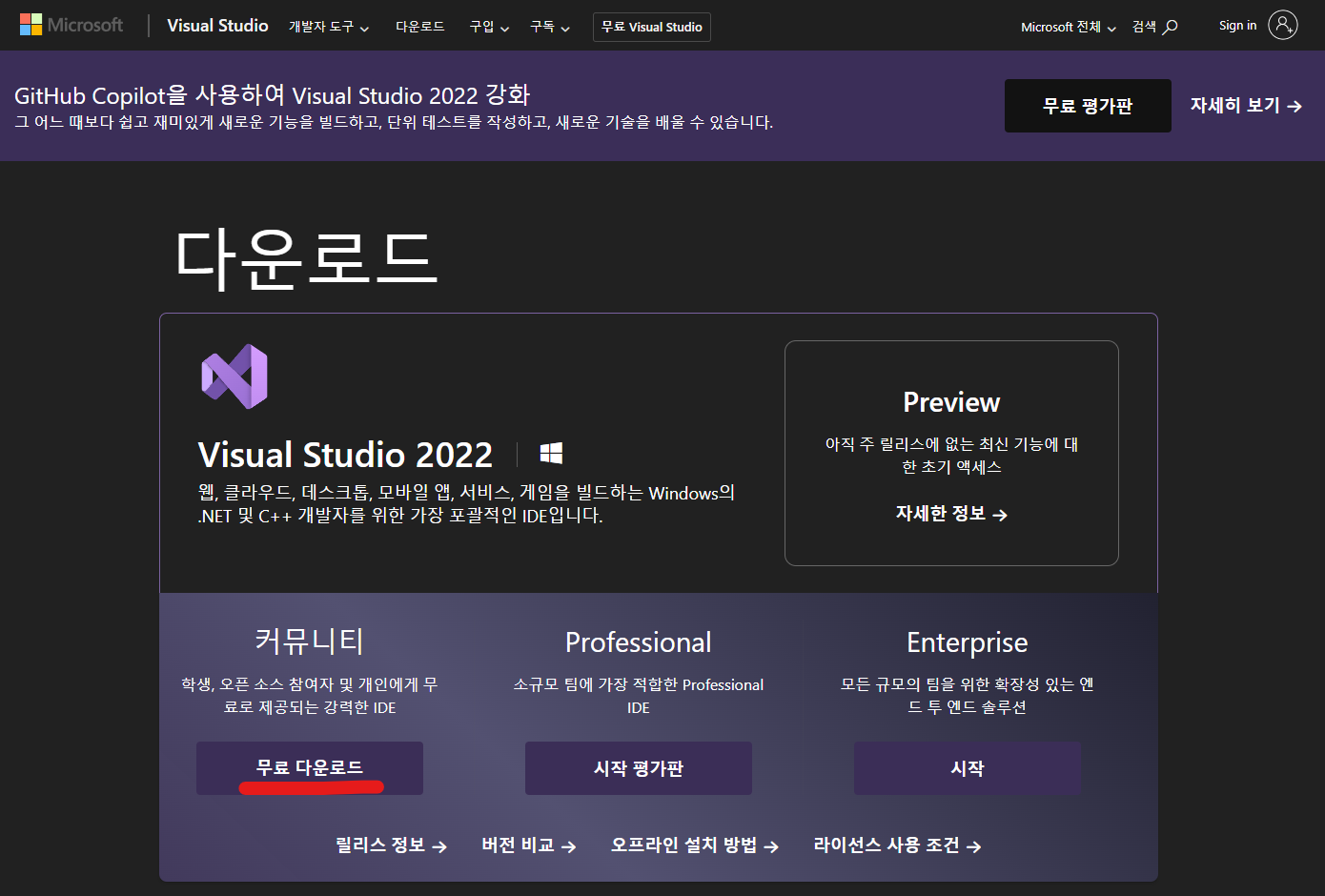
이 실습에는 2가지 Resouces를 활용합니다.

* Extension Training.oap 파일: Reactive Web App 모듈이 빌드된 OAP 파일
* Sample Code.txt 파일: MPSolver 액션에 붙일 코드가 있는 TXT 파일

## 2.2 시작하기

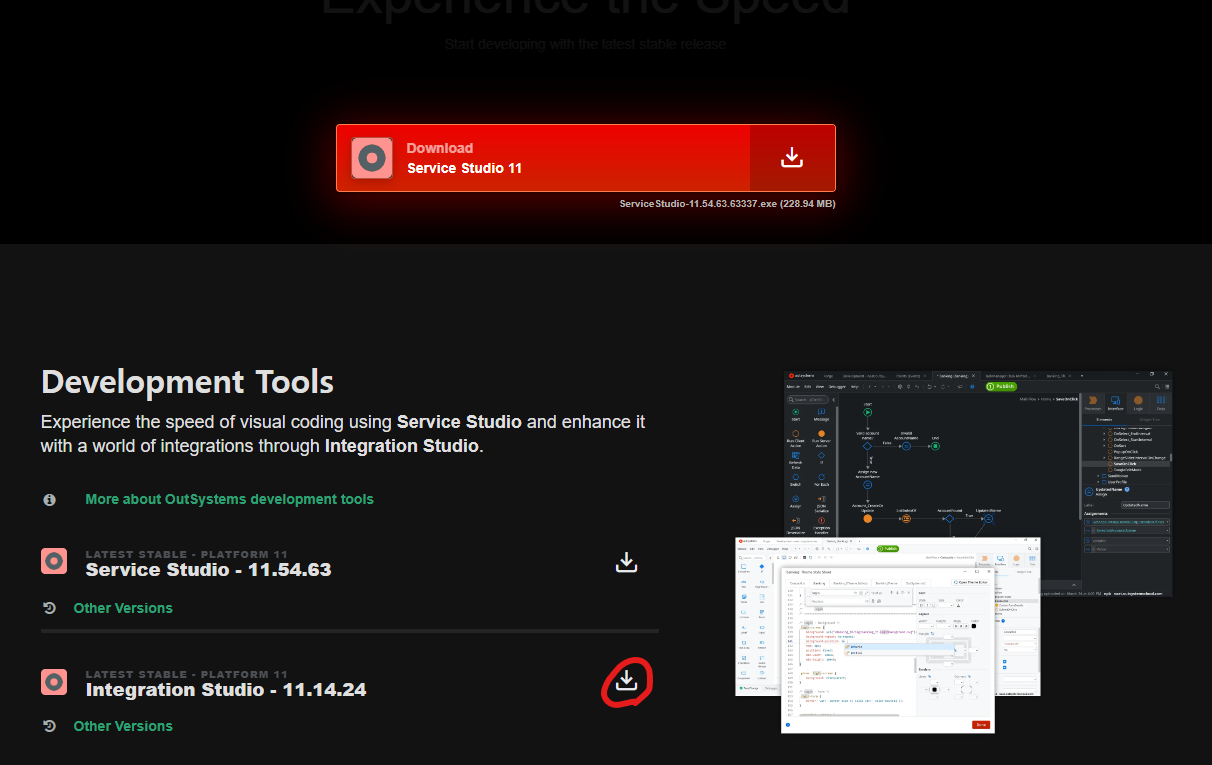
### 2.2.1 비주얼 스튜디오 (Visual Studio) 설치

C# 코드 수정 및 빌드를 하기 위해 [Microsoft의 Visual Studio 2022](https://visualstudio.microsoft.com/ko/downloads/)를 설치합니다.



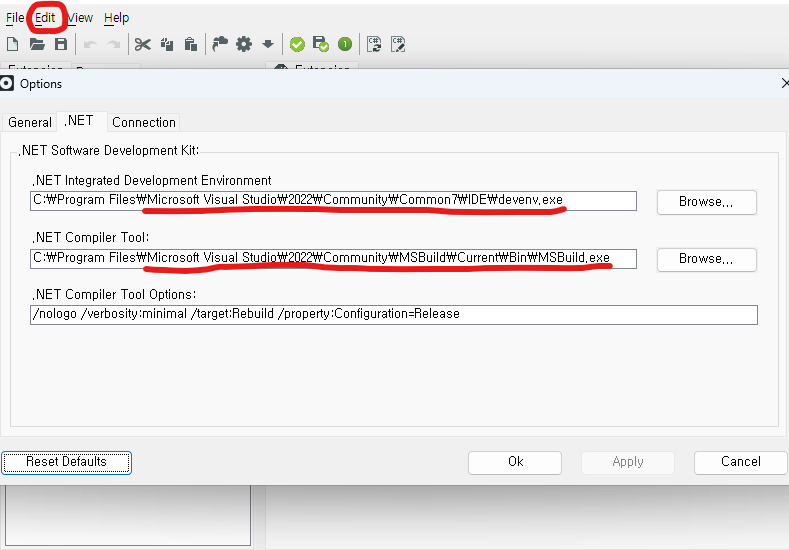
### 2.2.2 인티그레이션 스튜디오 (Integration Studio) 설치

Extension 모듈을 생성 및 수정하고, OutSystems와 .NET Framework 간의 연결을 쉽게 하기 위해 [OutSystems의 Integration Studio](https://www.outsystems.com/downloads/)를 설치합니다.



Integration Studio를 실행하여 Edit > Options > .NET 으로 접근하고,

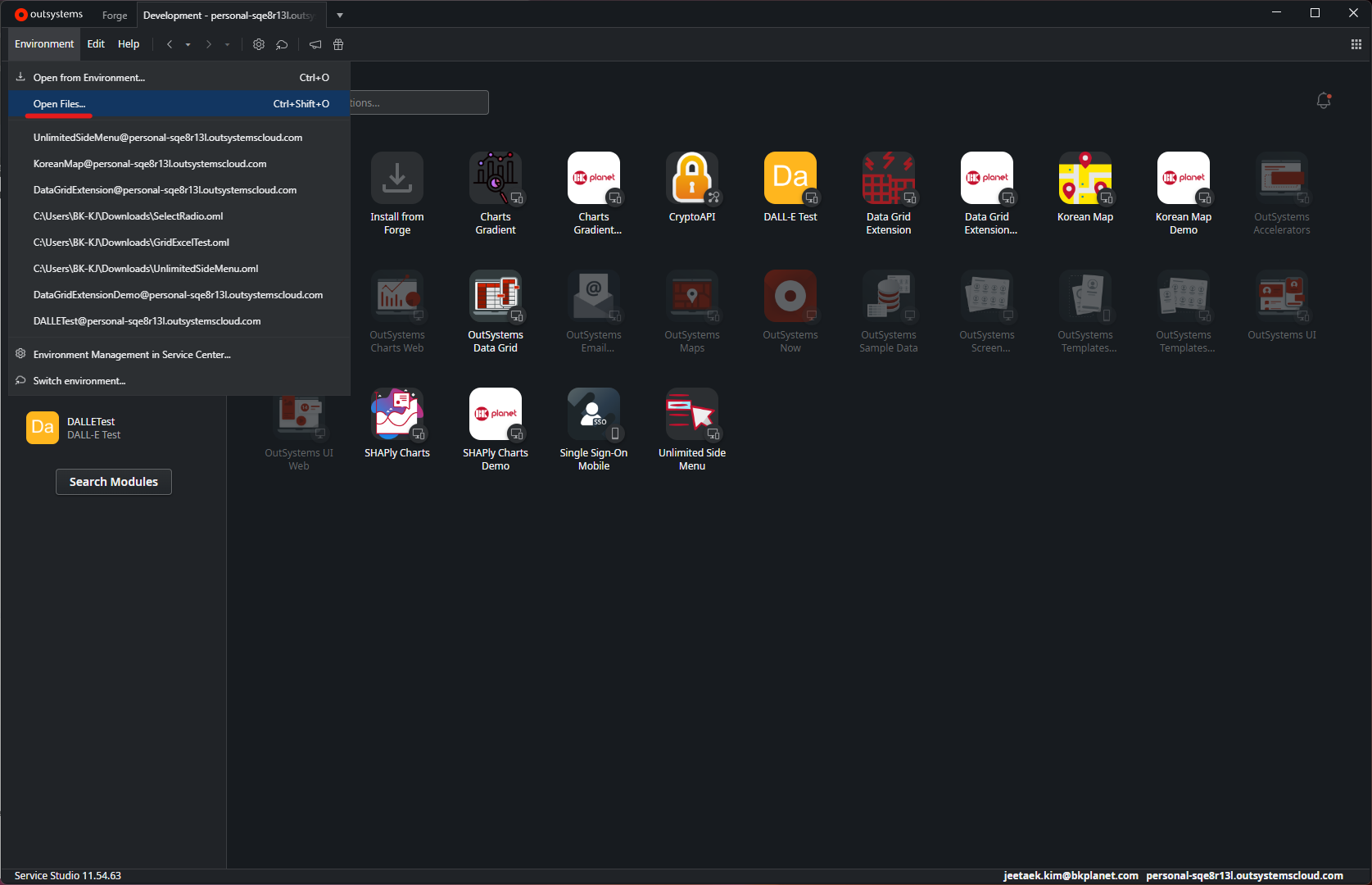
앞서 설치한 Visual Studio의 경로로 설정합니다.



### 

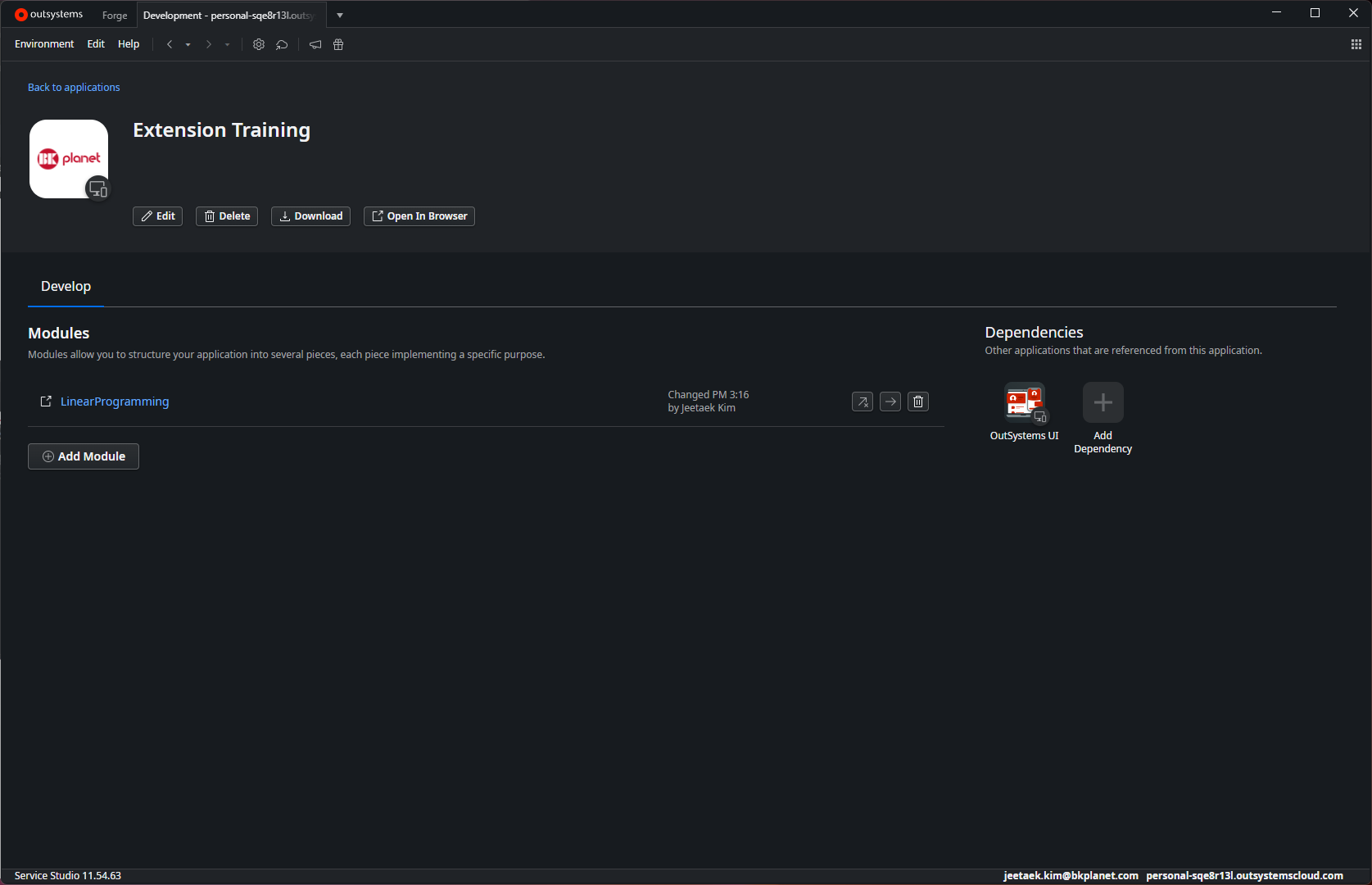
### 2.2.3 서비스 스튜디오 (Service Studio) 사용

교육과정을 쉽게 진행하기 위해 제공된 OAP 파일로 앱을 생성합니다.

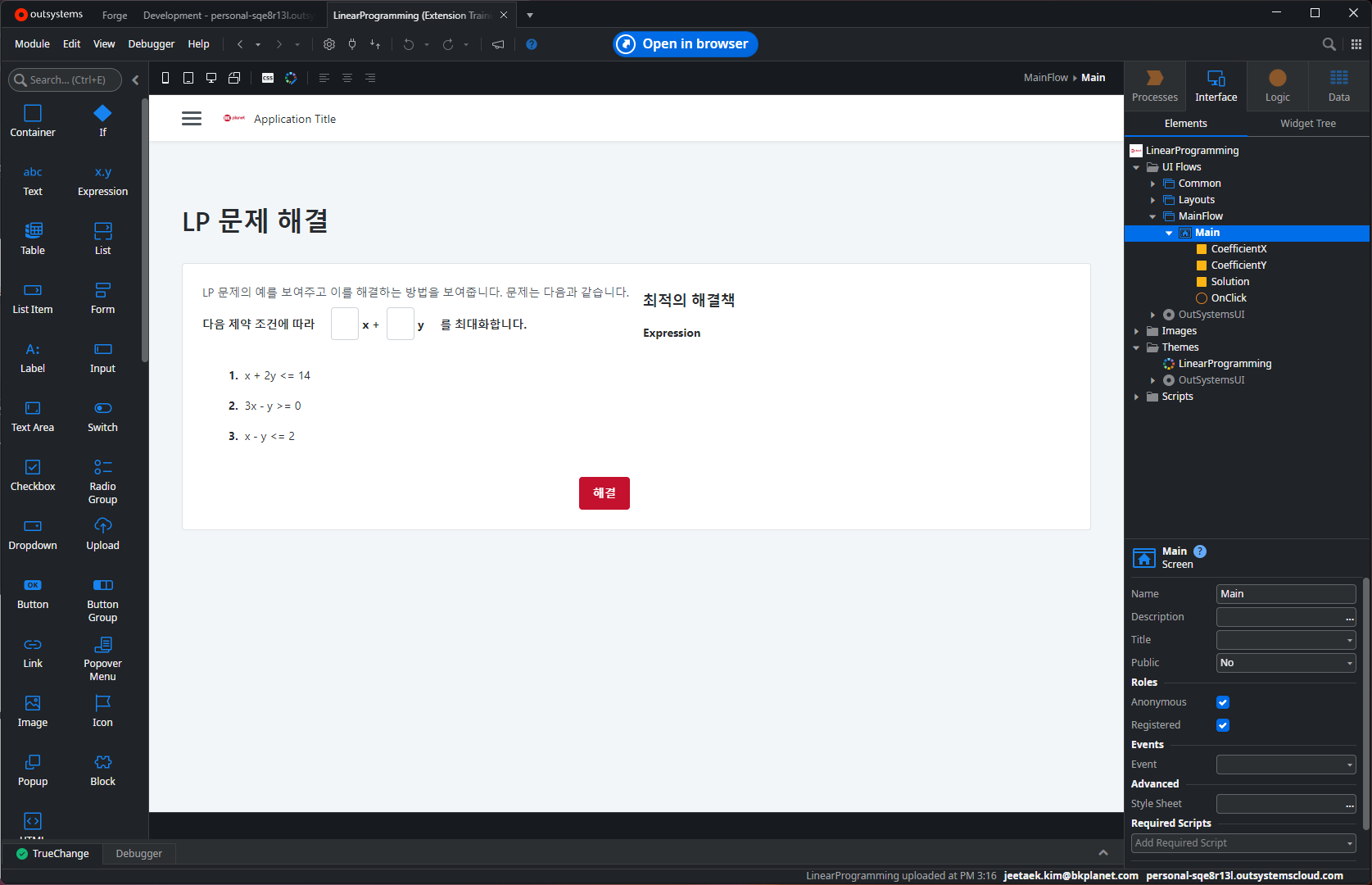


생성된 앱에는

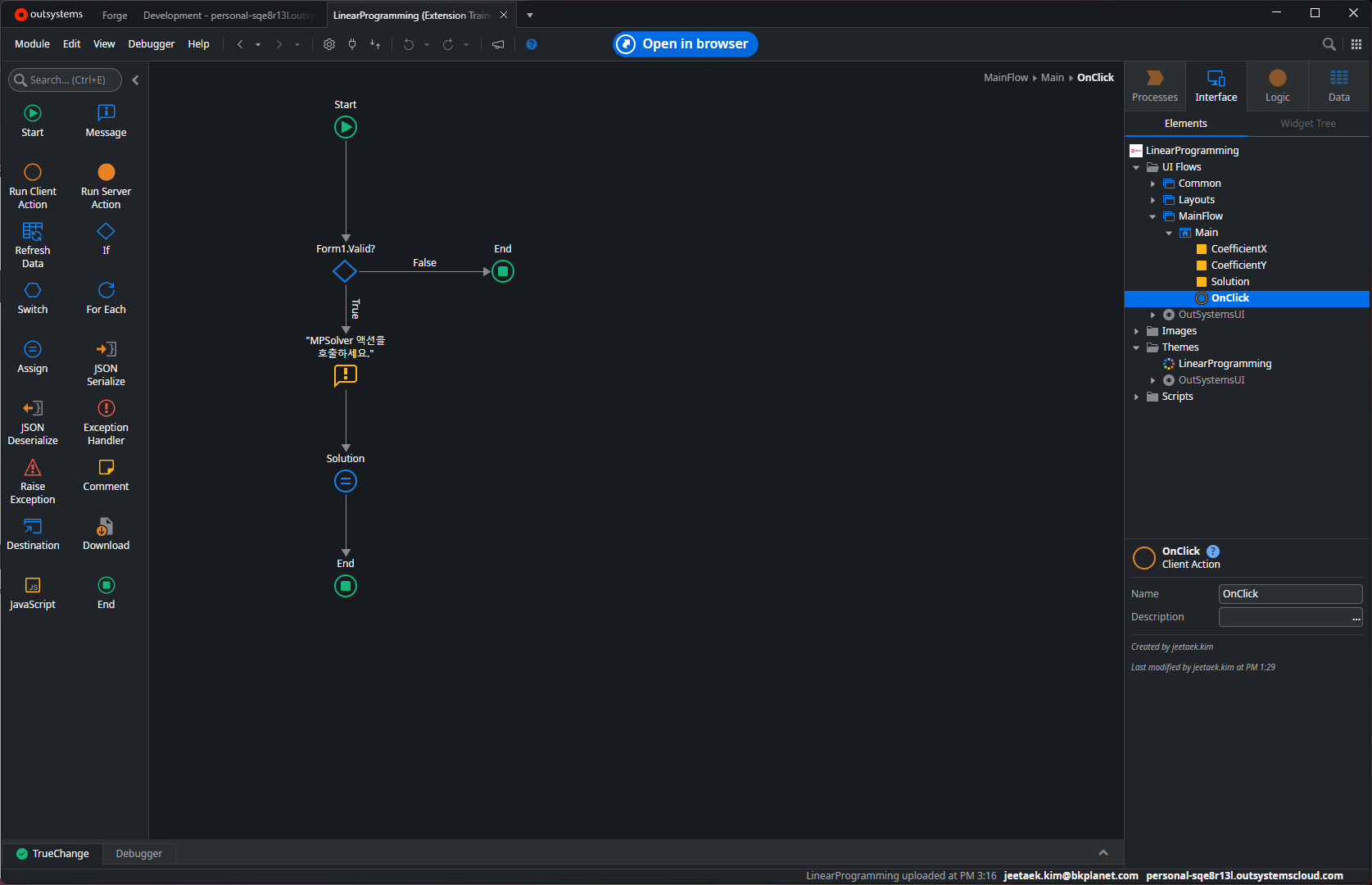
Extension 모듈을 호출할 Reactive Web App 모듈인 LinearProgramming,



브라우저에서 테스트할 수 있는 스크린인 Main,



C# 코드로 만든 함수를 호출할 액션인 OnClick이 미리 만들어져 있습니다.

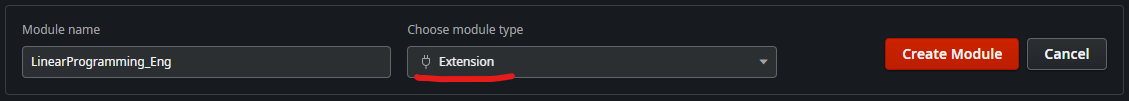
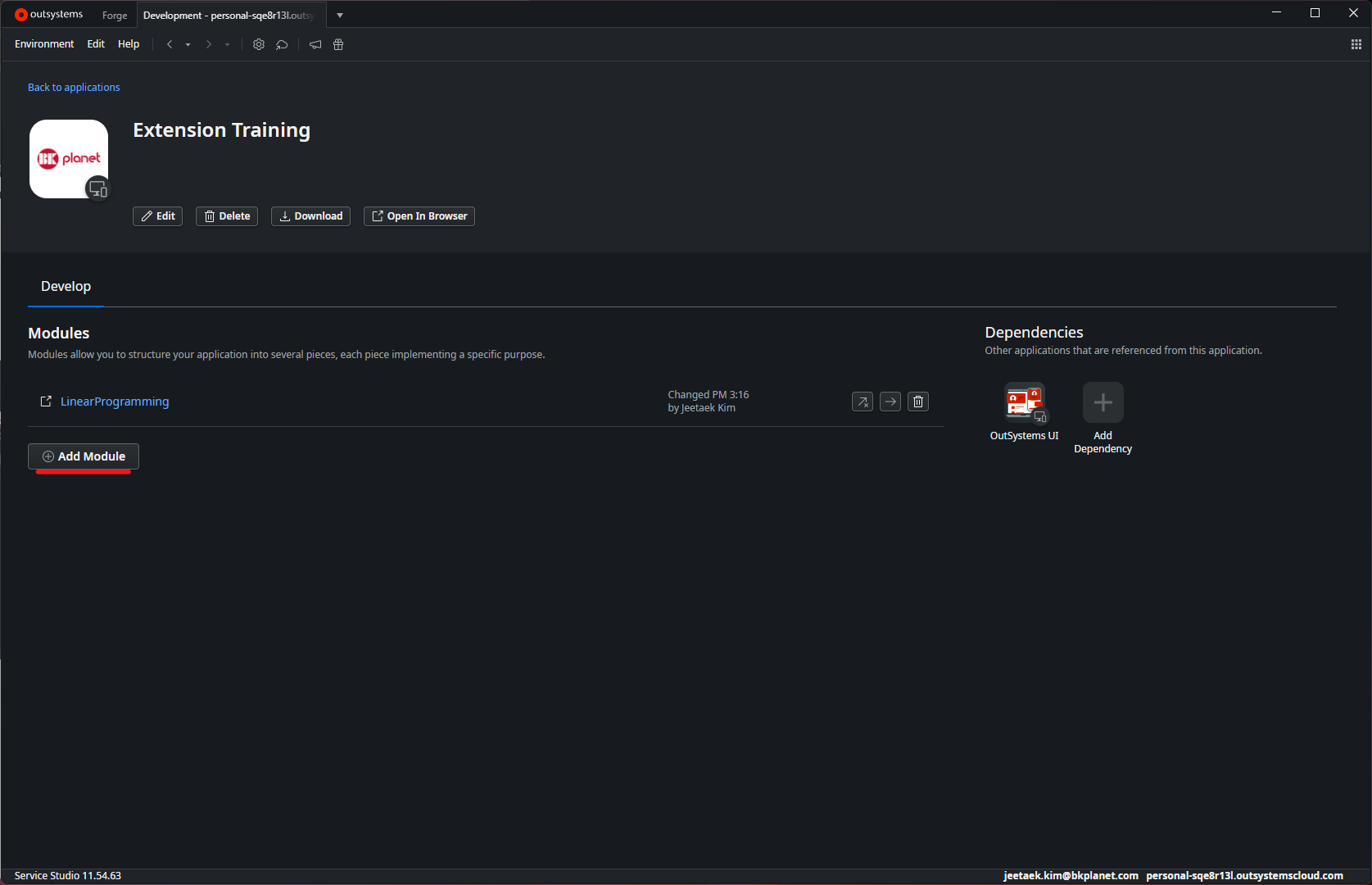


OAP 파일로 앱이 생성되지 않을 경우, Extension Training 및 LinearProgramming과 같은 이름의 앱과 모듈이 이미 존재하는지 확인하여 이름이 중복되지 않도록 하십시오.

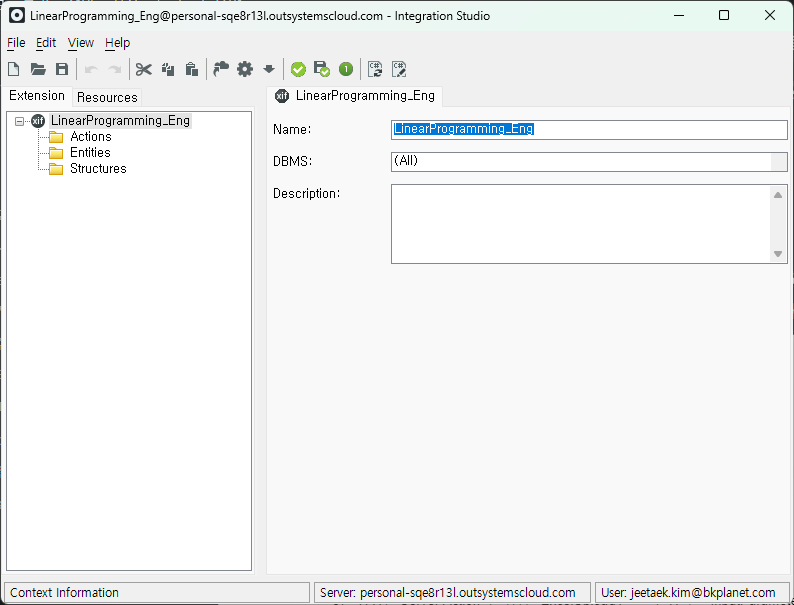
## 2.3 C# 코드 추가

이제 C# 코드를 빌드하는 방법을 실습하겠습니다.

1. Extension Training 앱에 LinearProgramming\_Eng라는 이름의 Extension 모듈을 추가합니다.

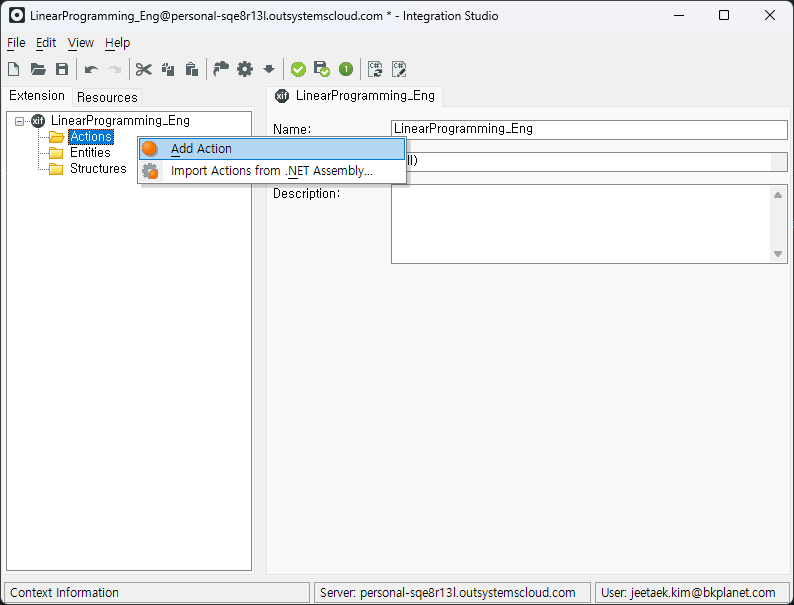


1. 자동으로 열린 Integration Studio에서 모듈이 추가될 환경으로 접속합니다.

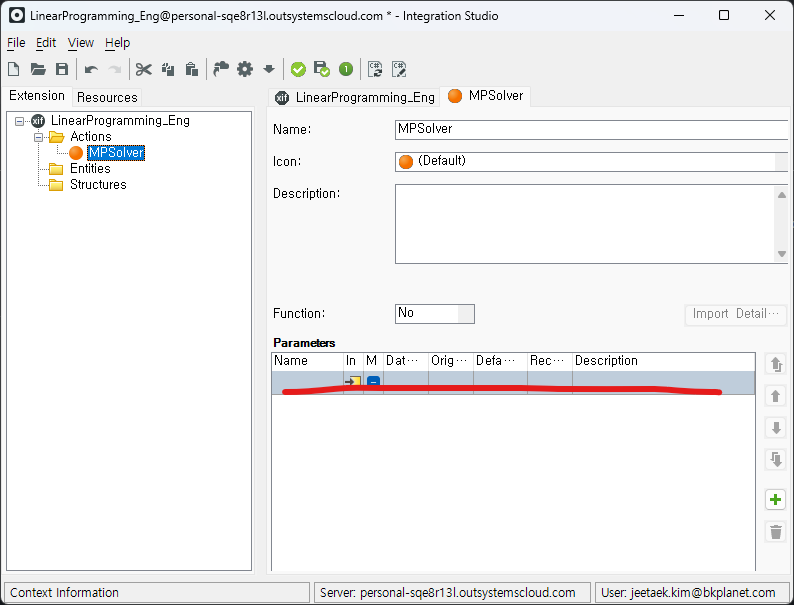


Username과 Password는 해당 환경에 Service Studio로 접속할 때와 동일합니다.

1. Actions 폴더를 우클릭하고 MPSolver라는 이름의 액션을 추가합니다.

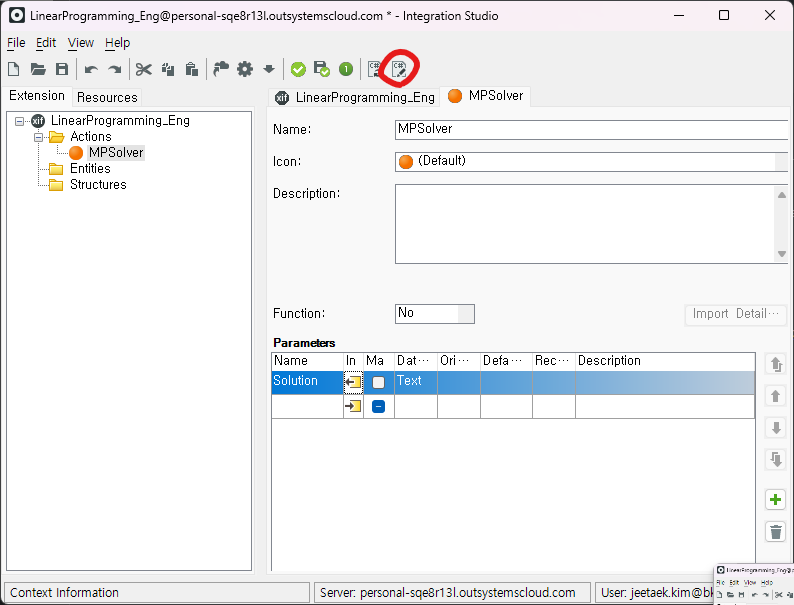


1. Parameters 영역을 클릭하여 Solution이라는 이름의, Text 타입인 출력변수를 추가합니다.

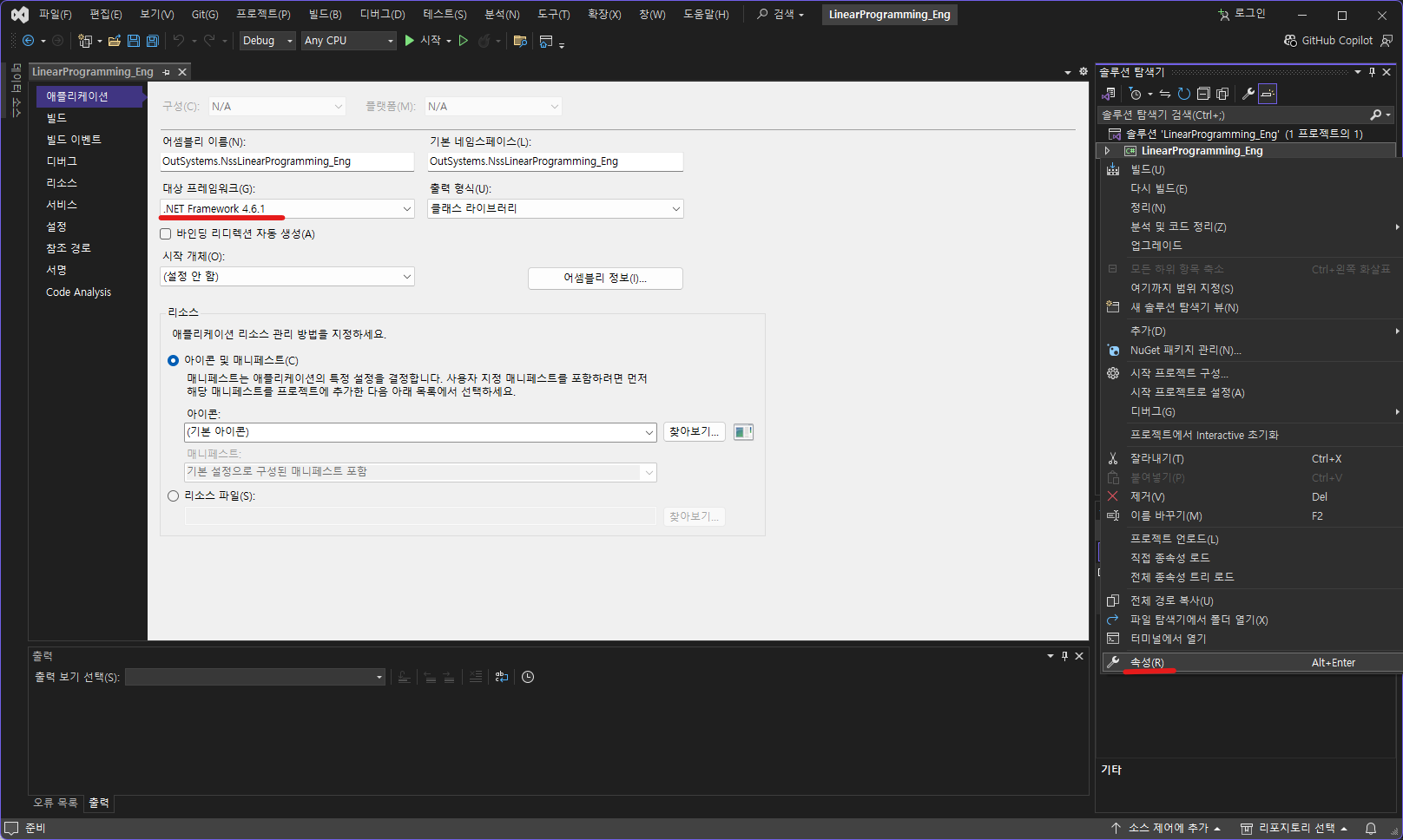


파라미터는 디폴트인 입력변수로 추가됩니다. 노란 박스 아이콘을 클릭하여 화살표가 박스에서 나가는 모양으로 변경하십시오.

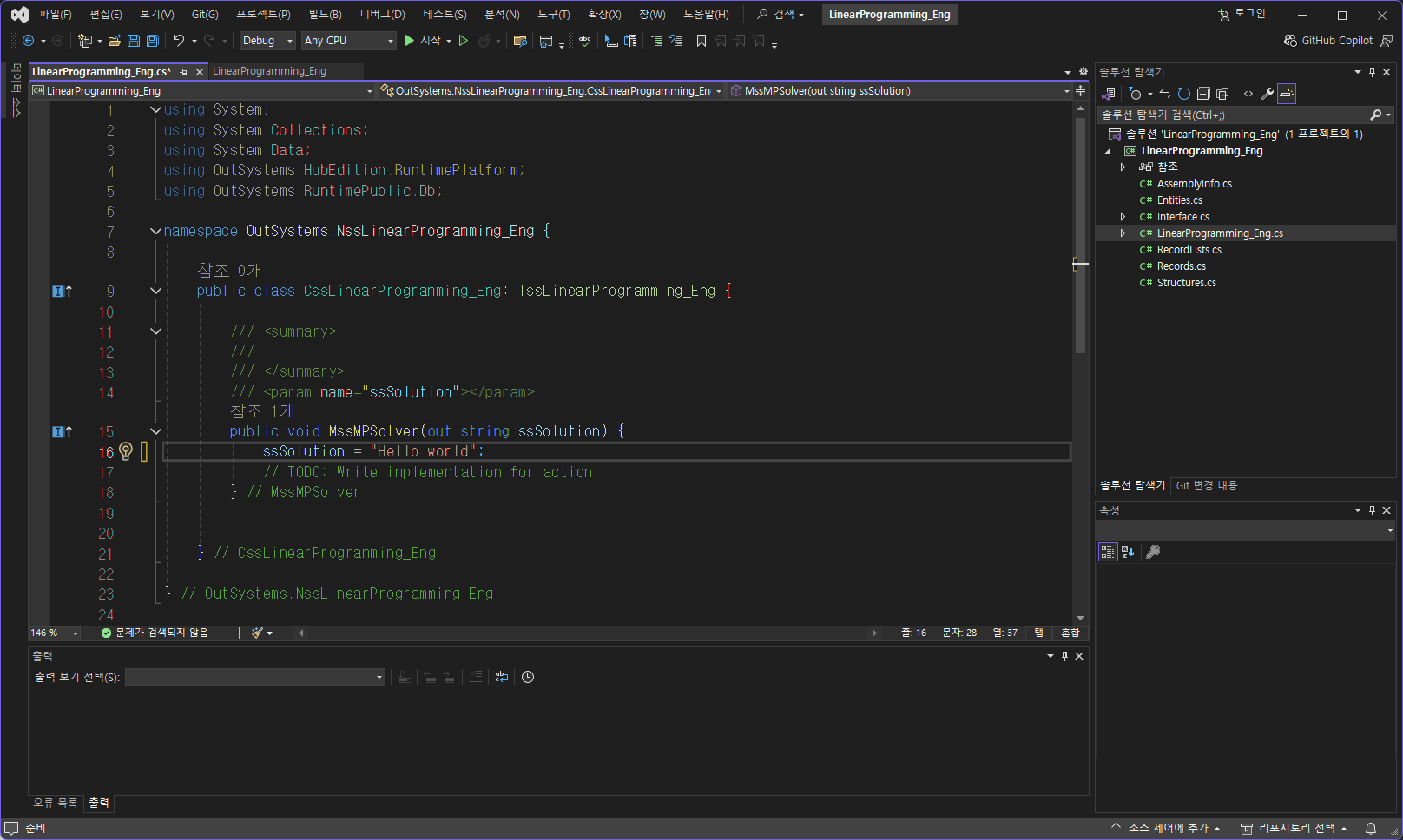
1. 아래 사진과 같은 형태가 되었으면, 툴바의 가장 오른쪽 아이콘인 Edit Source Code .NET을 클릭하여 Visual Studio로 이동합니다.



1. 솔루션의 속성에서 대상 프레임워크가 .NET Framework 4.6.1 버전인지 확인합니다.

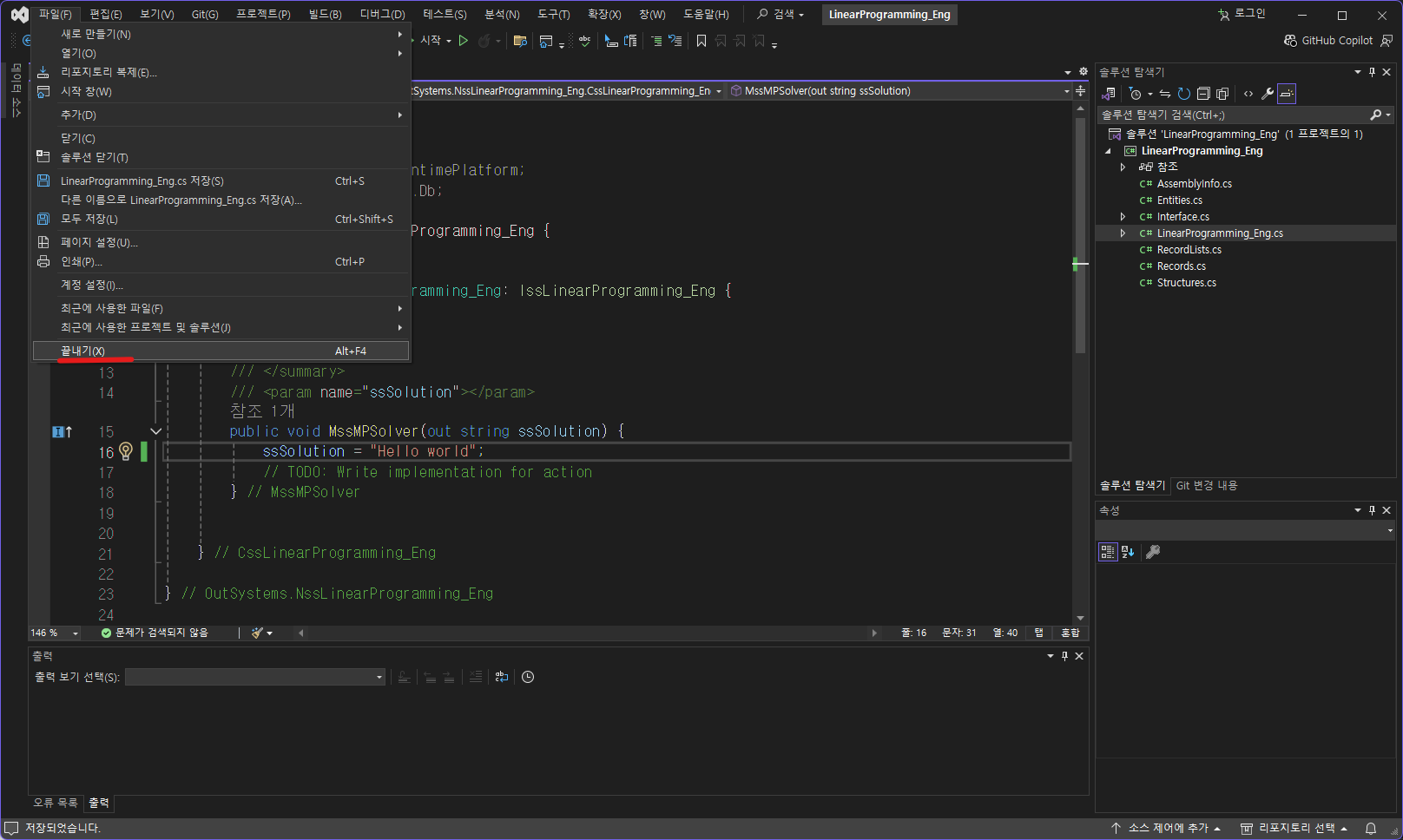


1. 프로젝트를 펼쳐 LinearProgramming\_Eng.cs 파일을 열고 앞서 Integration Studio에서 생성한 액션과 같은 MPSolver 함수를 확인 후 출력변수에 Hello world를 할당합니다.



함수명의 Mss 및 변수명의 ss 등은 OutSystems가 해석하기 위한 것이므로 수정하지 마십시오.

1. Ctrl+S 커맨드로 CS 파일을 저장하고, 파일 > 끝내기 로 Visual Studio를 종료합니다.

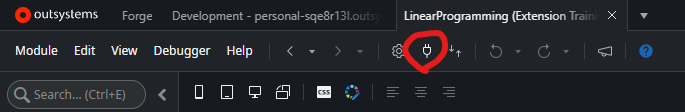


Visual Studio가 완전히 종료되지 않으면 다음 과정에서 에러가 뜰 수 있습니다.

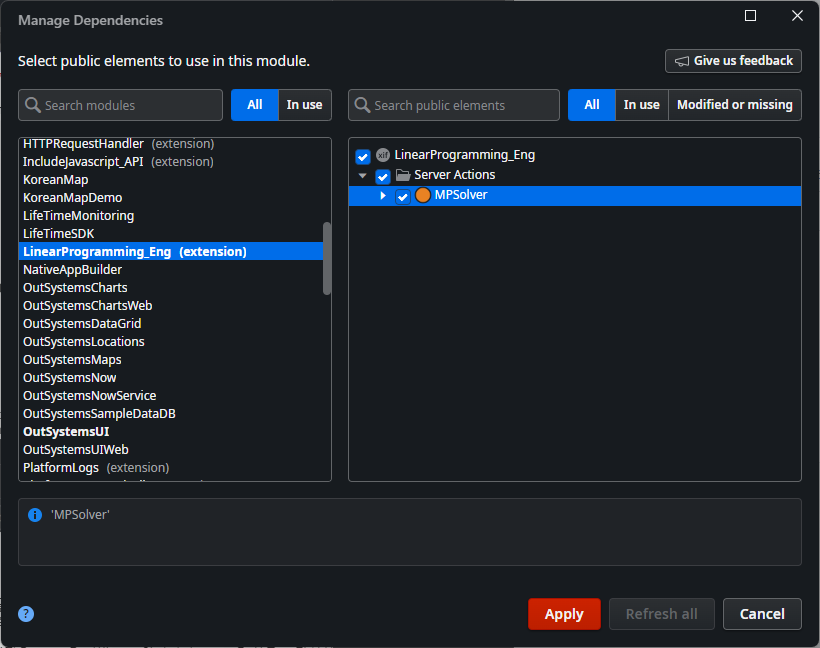
1. Integration Studio에서 Extension 모듈을 Publish 하고, Integration Studio를 종료합니다.



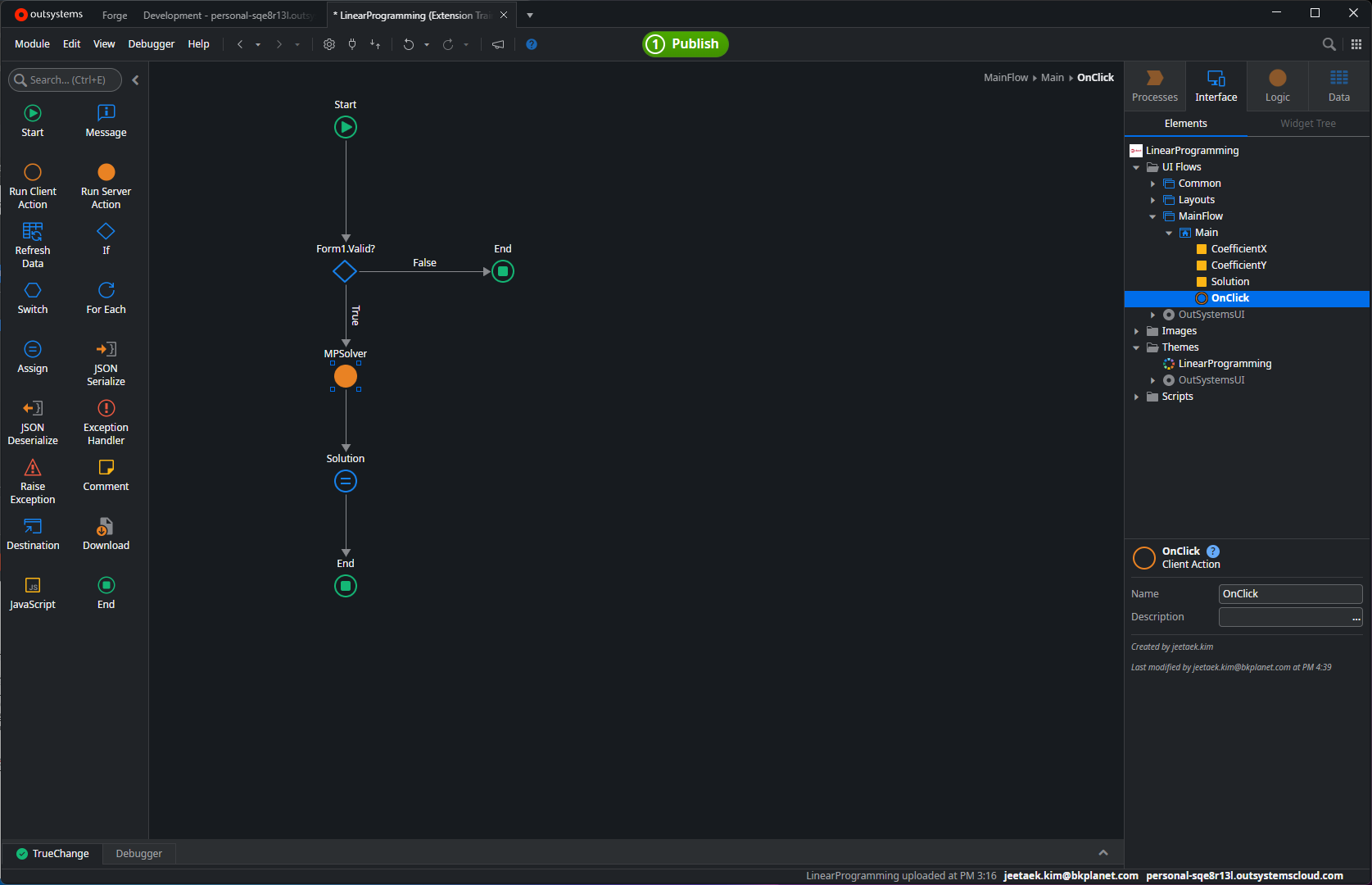
1. Service Studio로 이동하여 LinearProgramming 모듈에서 툴바의 플러그 아이콘인 Manage Dependencies…를 클릭합니다.



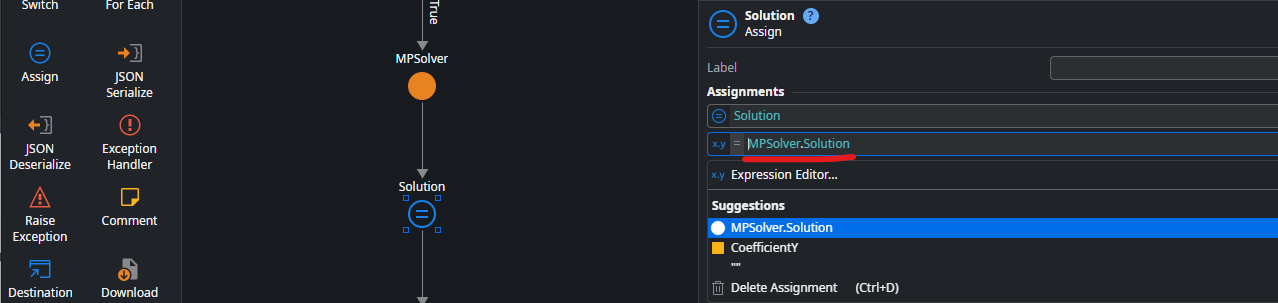
1. LinearProgramming\_Eng 모듈을 클릭하고 MPSolver 액션을 선택하여 Apply 합니다.



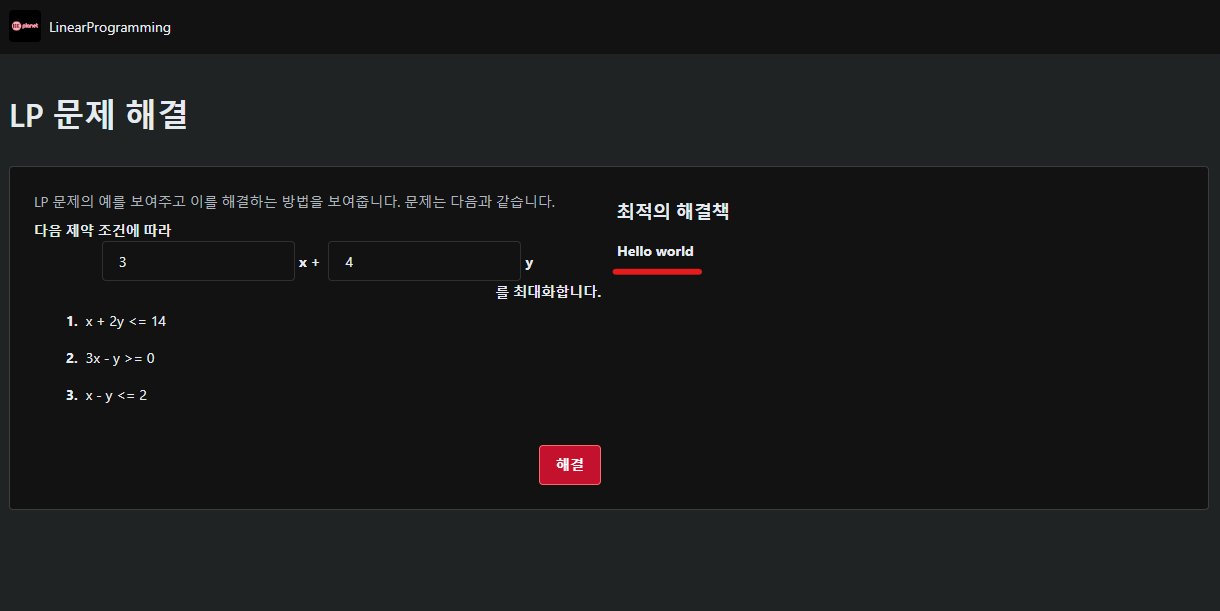
1. Main 스크린을 펼쳐 OnClick 액션을 더블클릭하여 접근 후, Message 노드 자리에 MPSolver 액션을 위치시킵니다.



1. MPSolver 액션 아래 Assign 노드를 클릭하여 Solution 변수에 MPSolver의 출력변수를 할당하고, 모듈을 Publish 합니다.



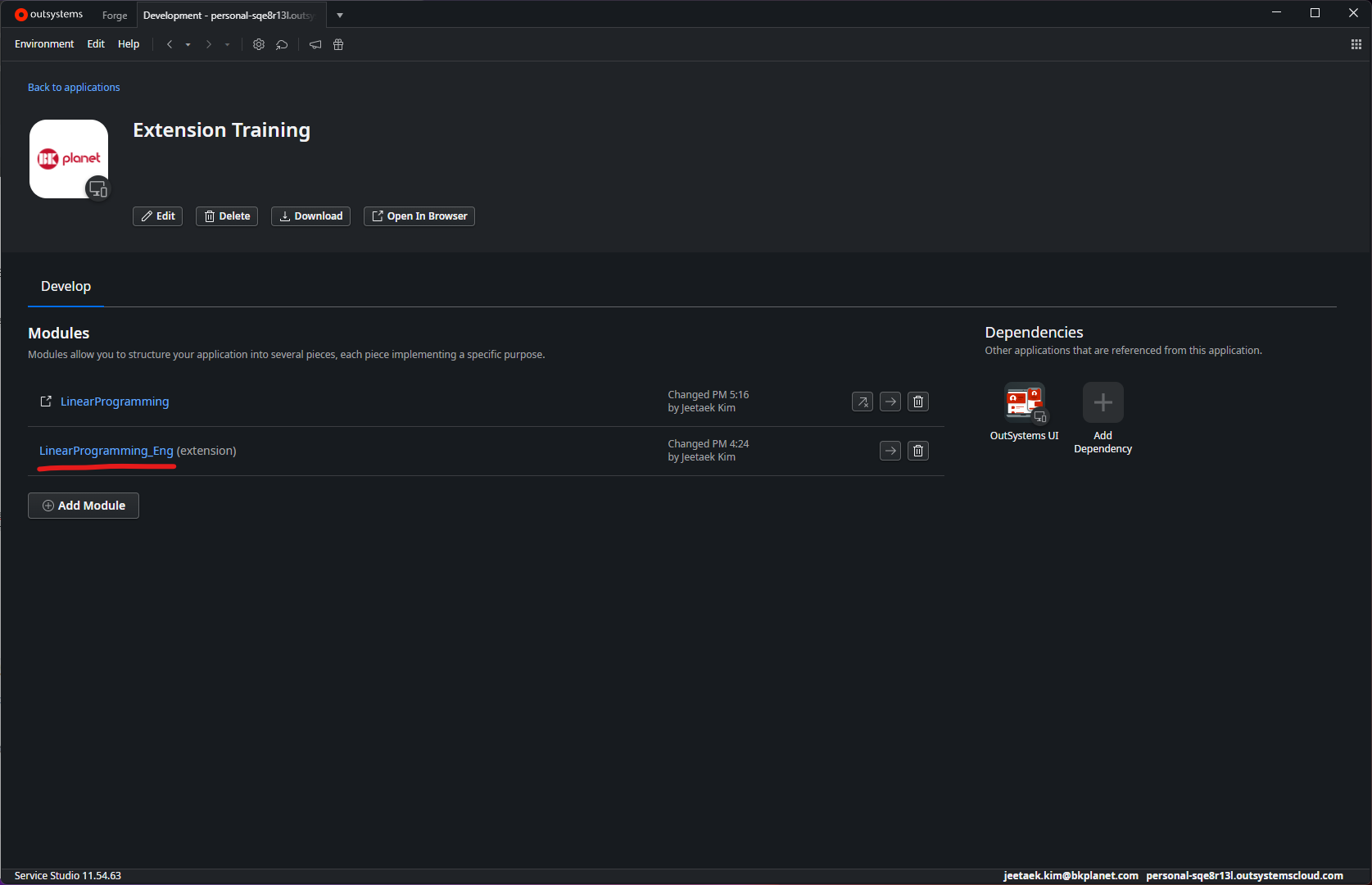
1. Publish가 끝나면 그 자리에 생긴 Open in browser 버튼을 클릭하여 브라우저로 이동하고, 해결 버튼을 클릭하여 Hello world가 출력되는지 확인합니다.



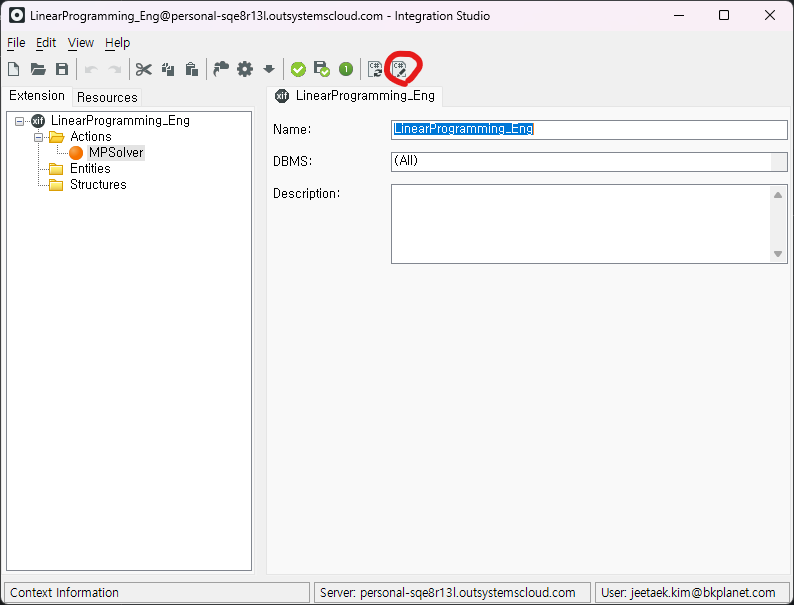
## 2.4 NuGet 패키지 추가

이제 Extension 모듈에 Google의 OR-Tools를 빌드하는 방법을 실습하겠습니다.

1. Service Studio에서 LinearProgramming\_Eng 모듈을 클릭하여 Integration Studio로 이동합니다.

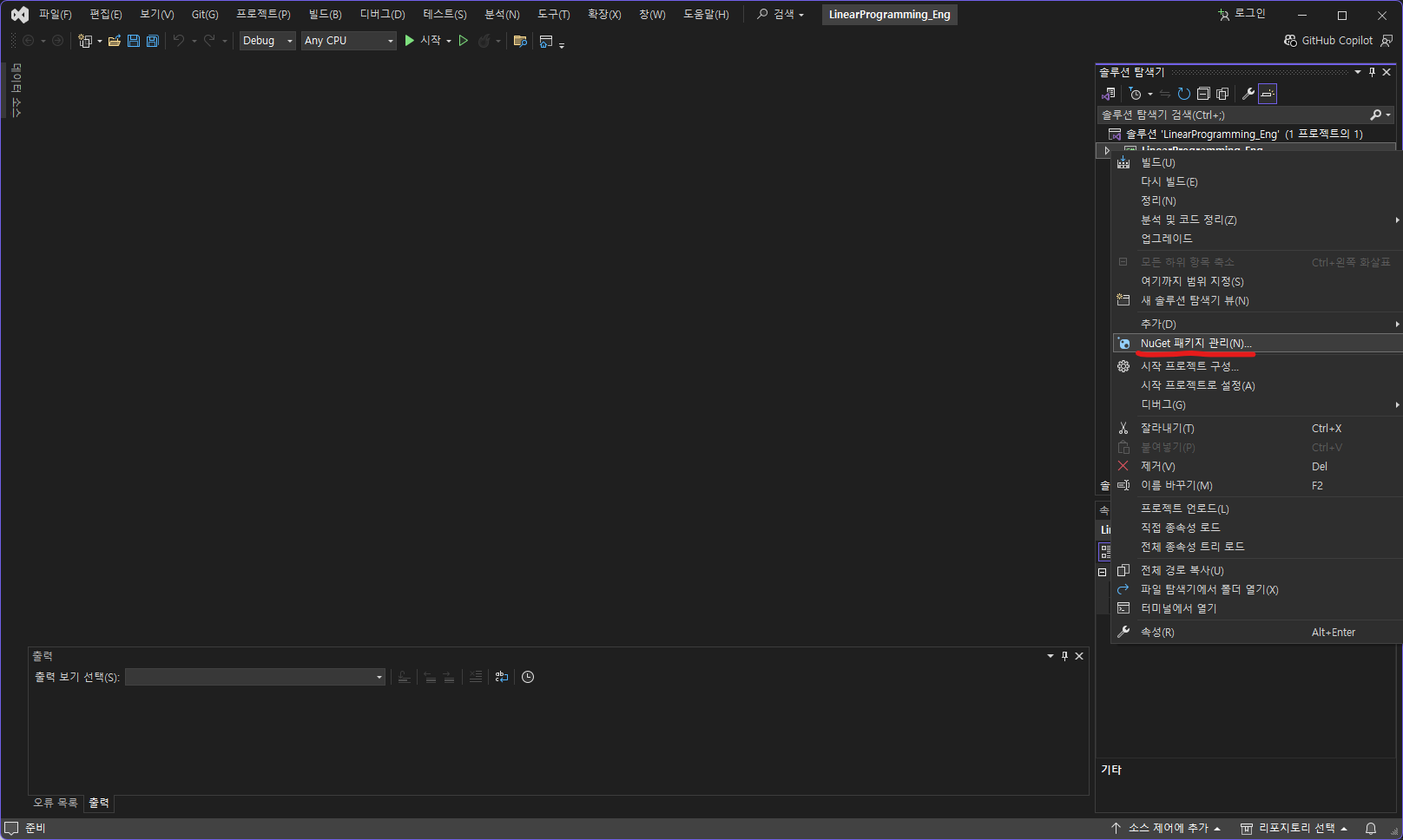


1. Integration Studio에서 툴바의 가장 오른쪽 아이콘을 클릭하여 Visual Studio로 이동합니다.

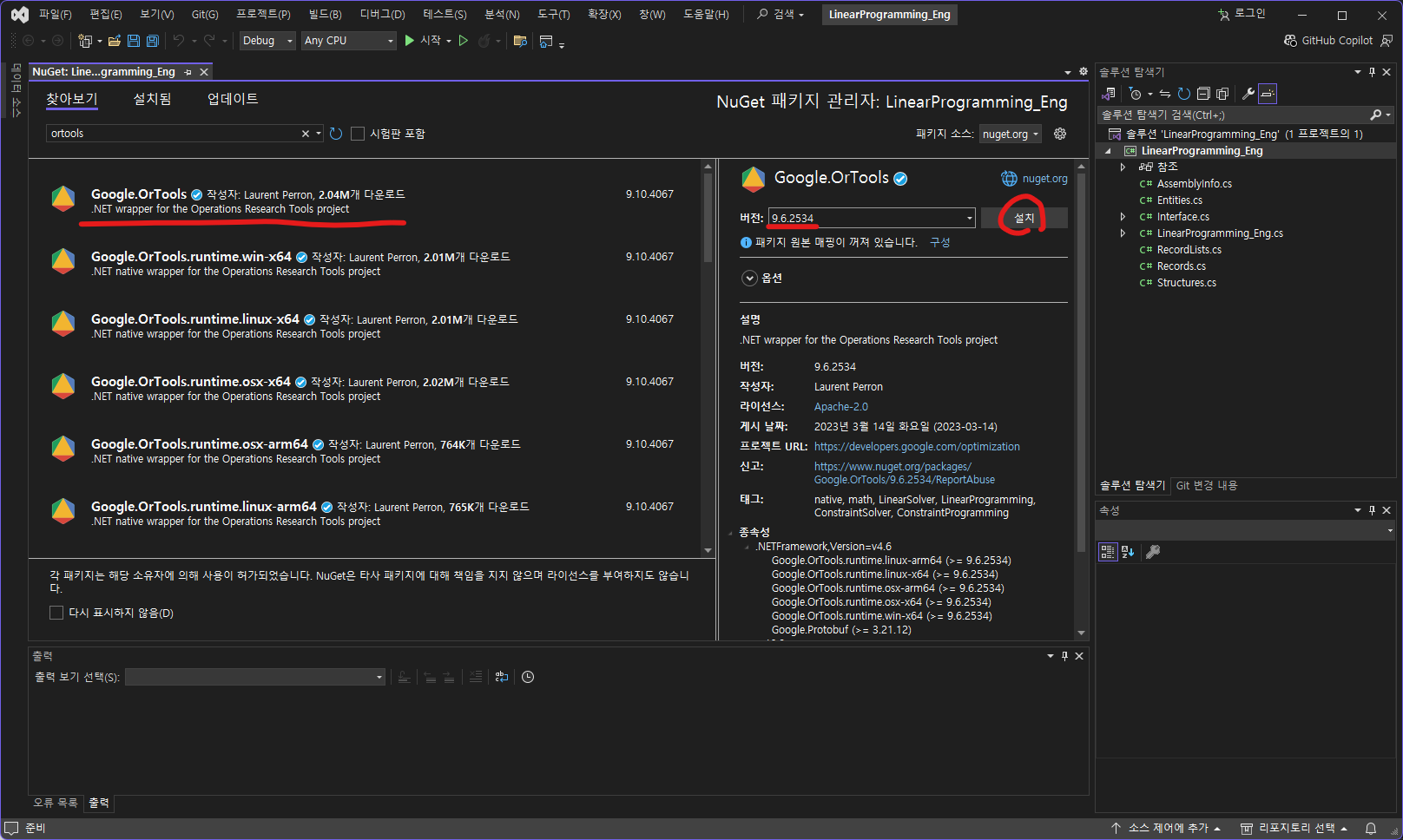


에러가 뜨는 경우, 해당 아이콘의 바로 왼쪽 아이콘인 Update Source Code를 클릭하여 Visual Studio로 이동하십시오.

1. 프로젝트를 우클릭하여 NuGet 패키지 관리를 클릭합니다.



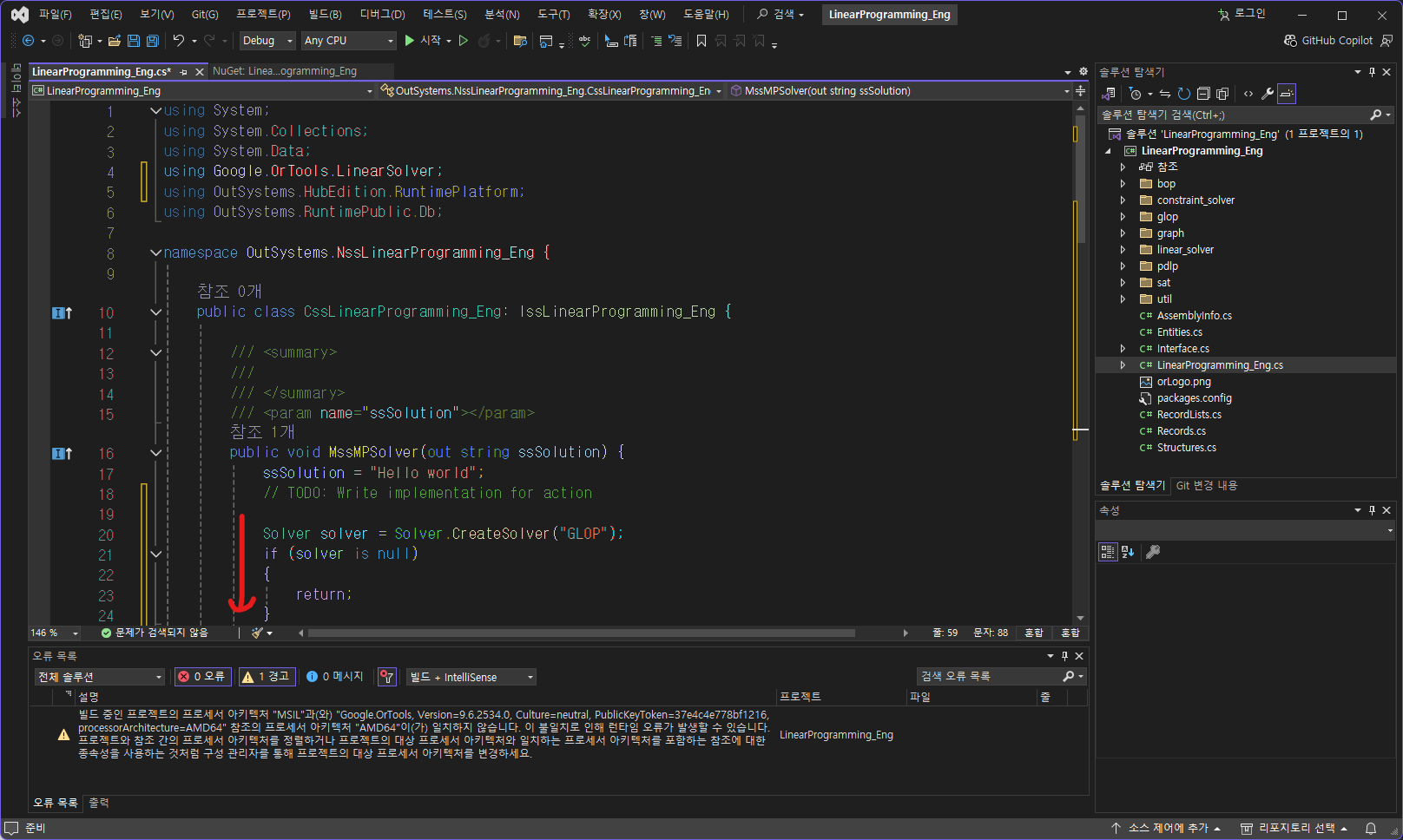
1. Google.OrTools를 검색하여 9.6.2534 버전을 설치합니다.



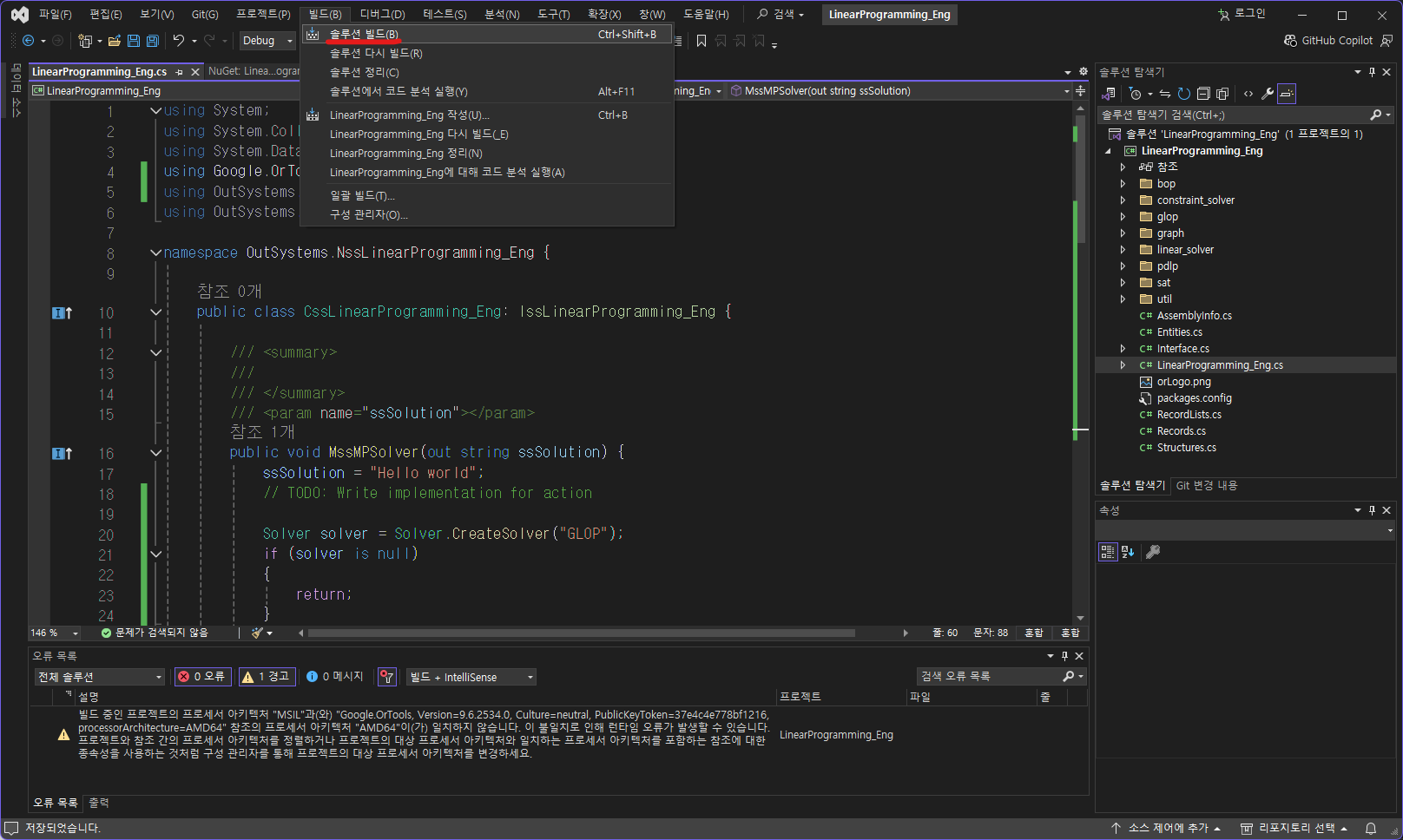
9.6.2534 버전이 .NET Framework 4.6.1에 호환되는 최신 버전입니다.

\*\* Personal Environment에 진행하는 경우, **9.2.9972 버전**으로 설치하십시오.

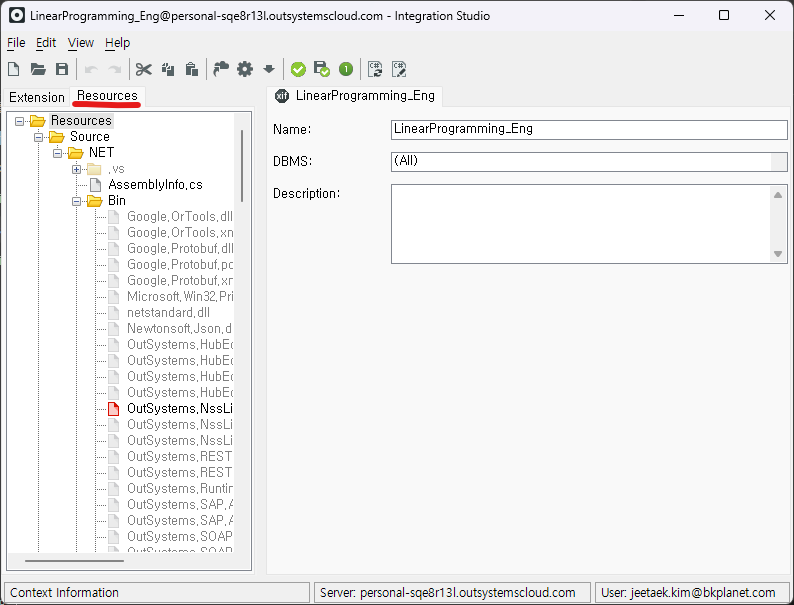
1. LinearProgramming\_Eng.cs 파일을 열고, MPSolver 함수 안의 주석 아래에 Sample Code.txt 내용을 붙여넣습니다.



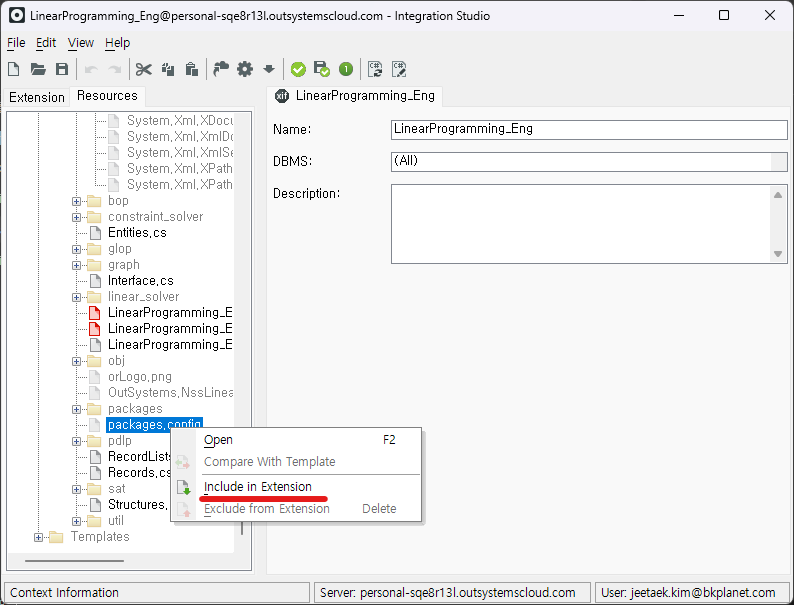
1. Ctrl+S 커맨드로 CS 파일을 저장하고, 솔루션 빌드를 합니다.



1. Visual Studio를 종료하고 Integration Studio로 이동하여 Resources 탭을 클릭합니다.



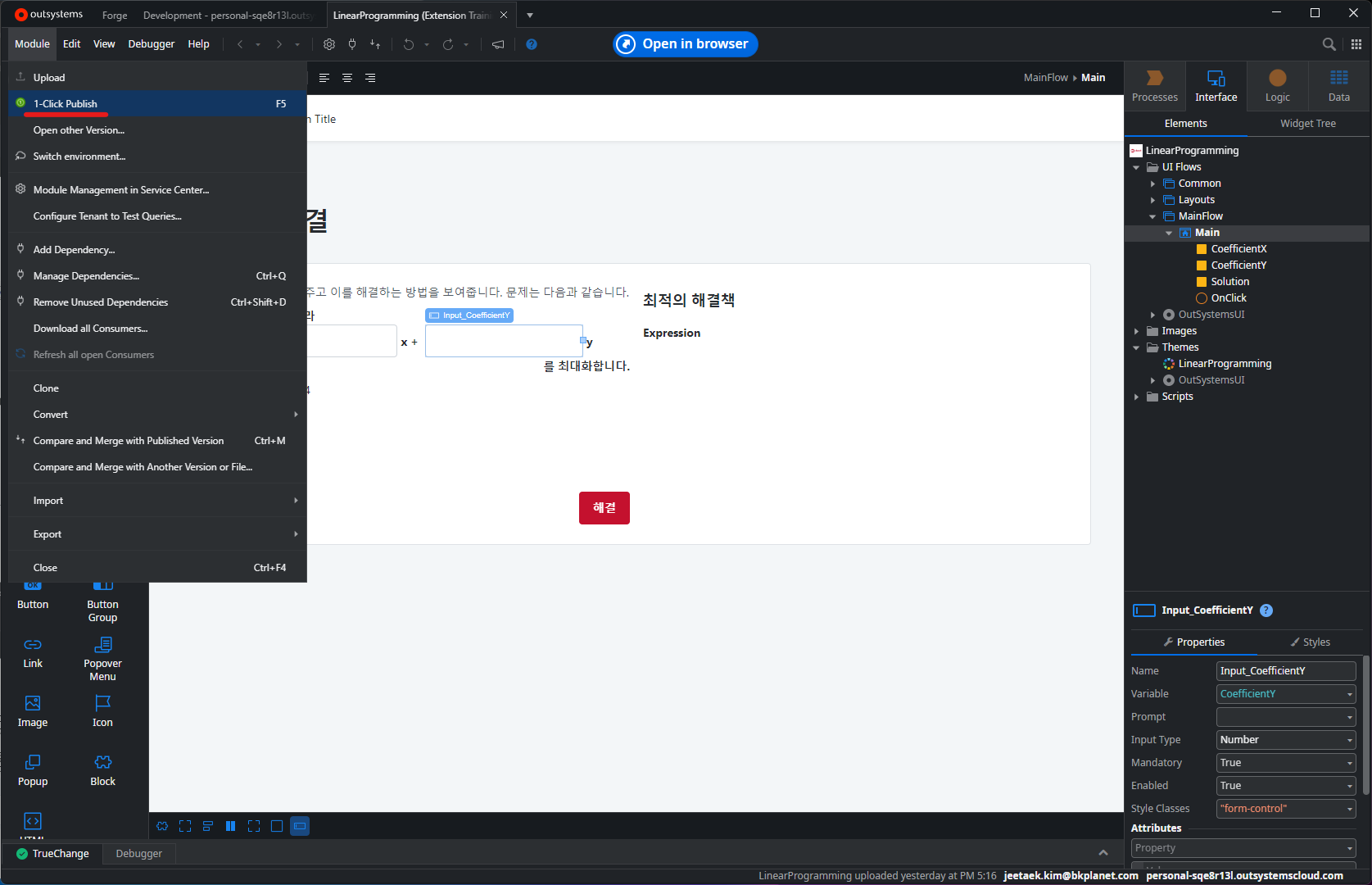
1. Resources > Source > NET 폴더 안의 packages.config 파일을 우클릭하여 모듈에 포함시킵니다.



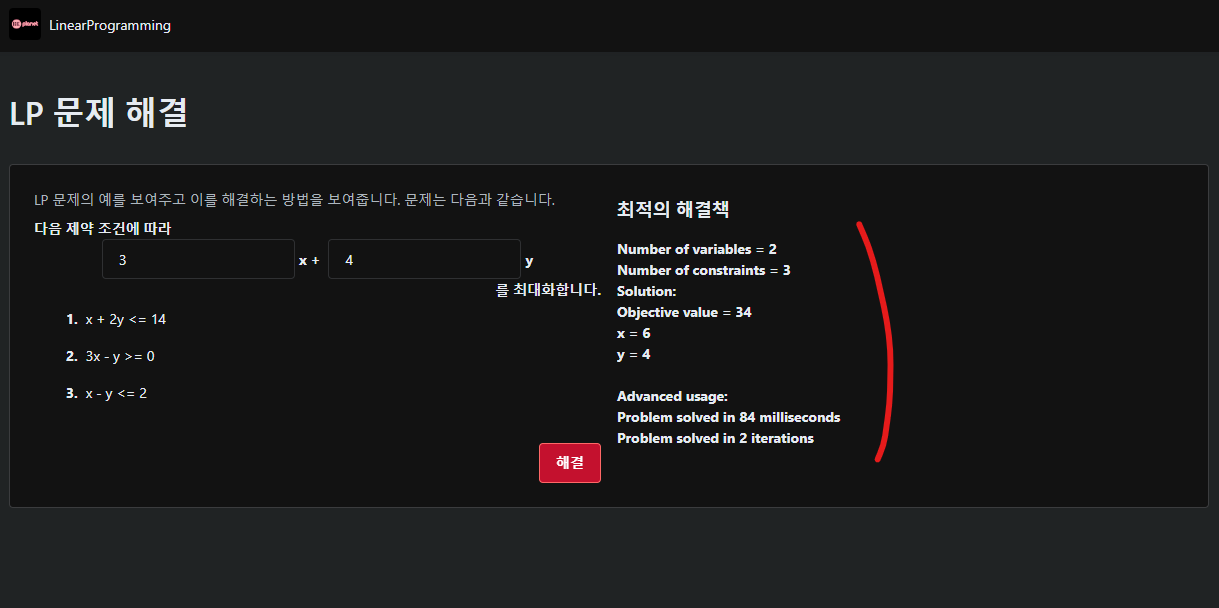
1. Integration Studio에서 Extension 모듈을 Publish 하고, Integration Studio를 종료합니다.



1. Service Studio로 이동하여 LinearProgramming 모듈에서 1-Click Publish를 합니다.



1. Publish가 끝나면 Open in browser 버튼을 클릭하여 브라우저로 이동하고, 해결 버튼을 클릭하여 수정한 결과가 출력되는지 확인합니다.

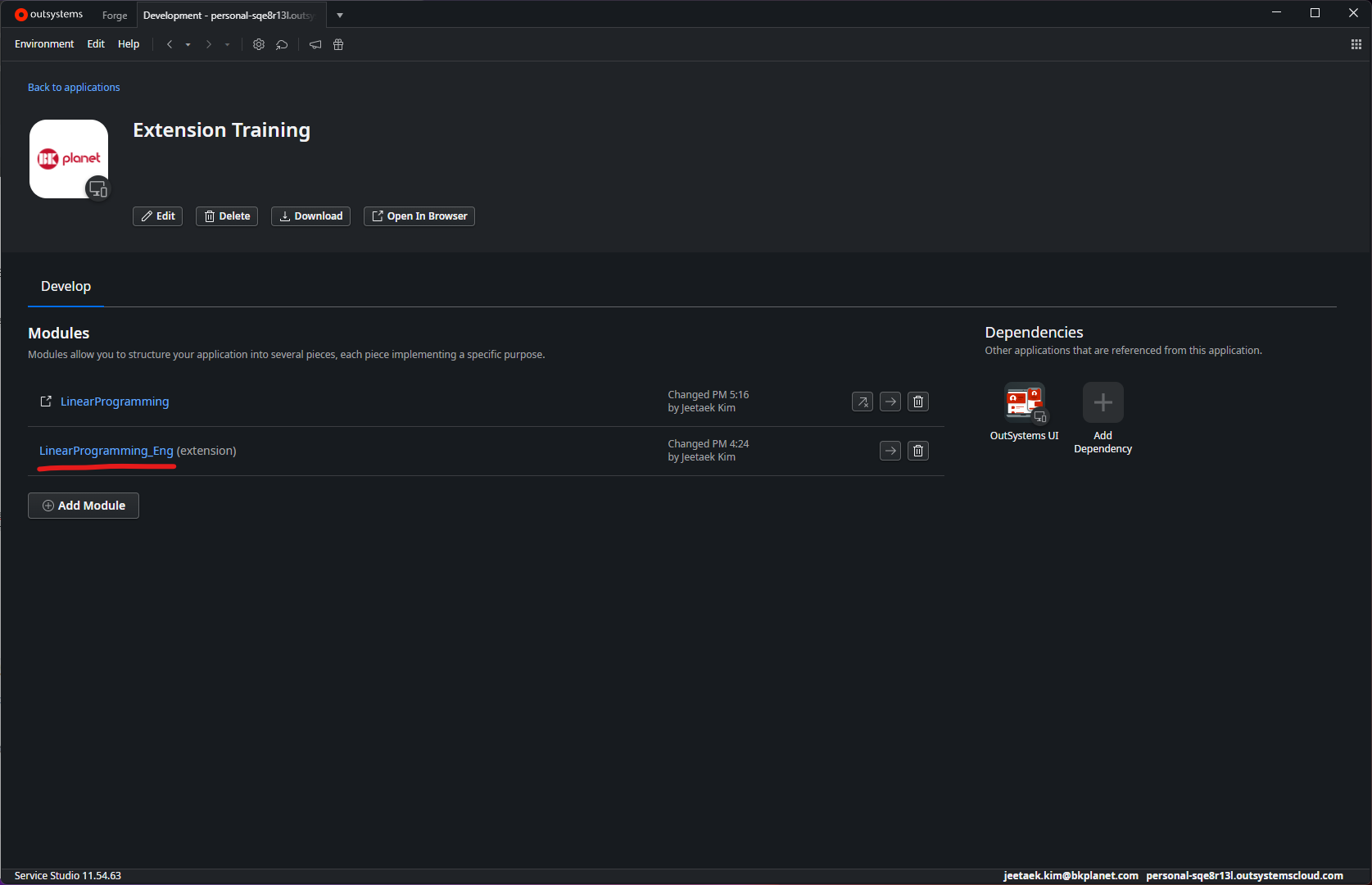


## 

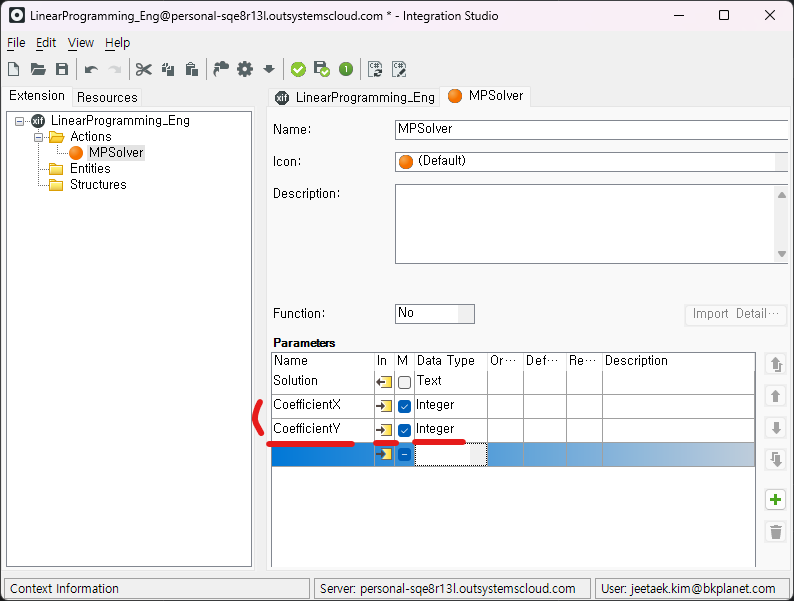
## 2.5 마무리하기

이제 사용자가 입력한 최대화 문제에 반응하는 선형 최적화 기능을 구축하겠습니다.

1. Service Studio에서 LinearProgramming\_Eng 모듈을 클릭하여 Integration Studio로 이동합니다.

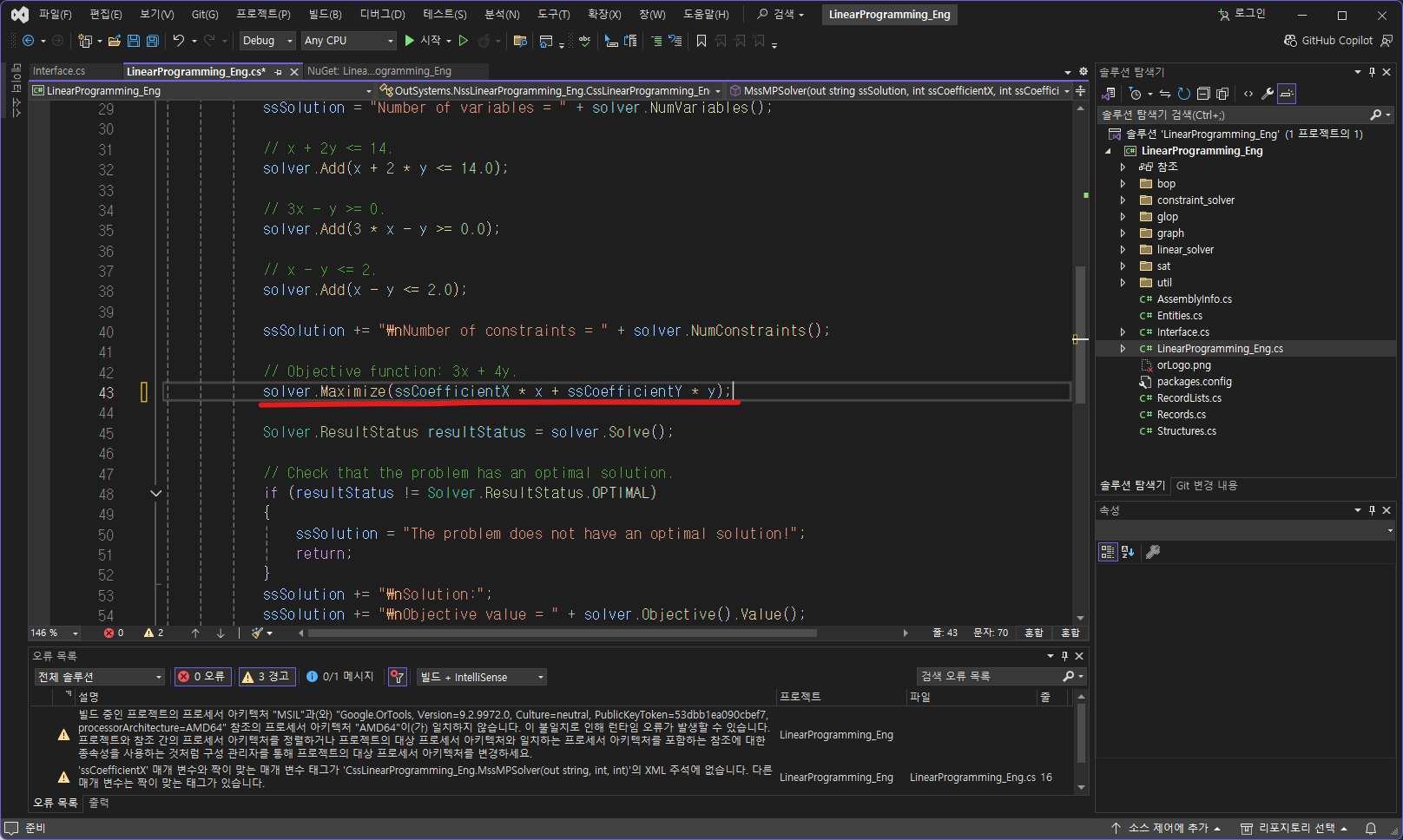


1. MPSolver를 더블클릭하여 열고, 입력변수 CoefficientX, CoefficientY를 추가합니다.



1. Visual Studio로 이동하여 LinearProgramming\_Eng.cs 파일을 열고, MPSolver 함수 안에서 Solver.Maximize 함수를 호출하는 부분을 찾아서 그 인자를 수정합니다.

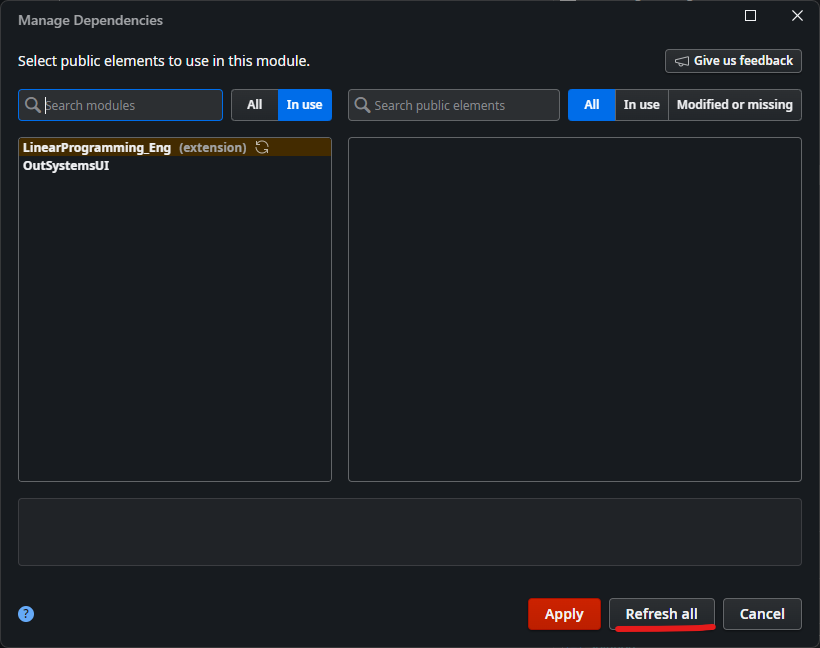
~~solver.Maximize(3 \* x + 4 \* y);~~ ===> solver.Maximize(ssCoefficientX \* x + ssCoefficientY \* y);



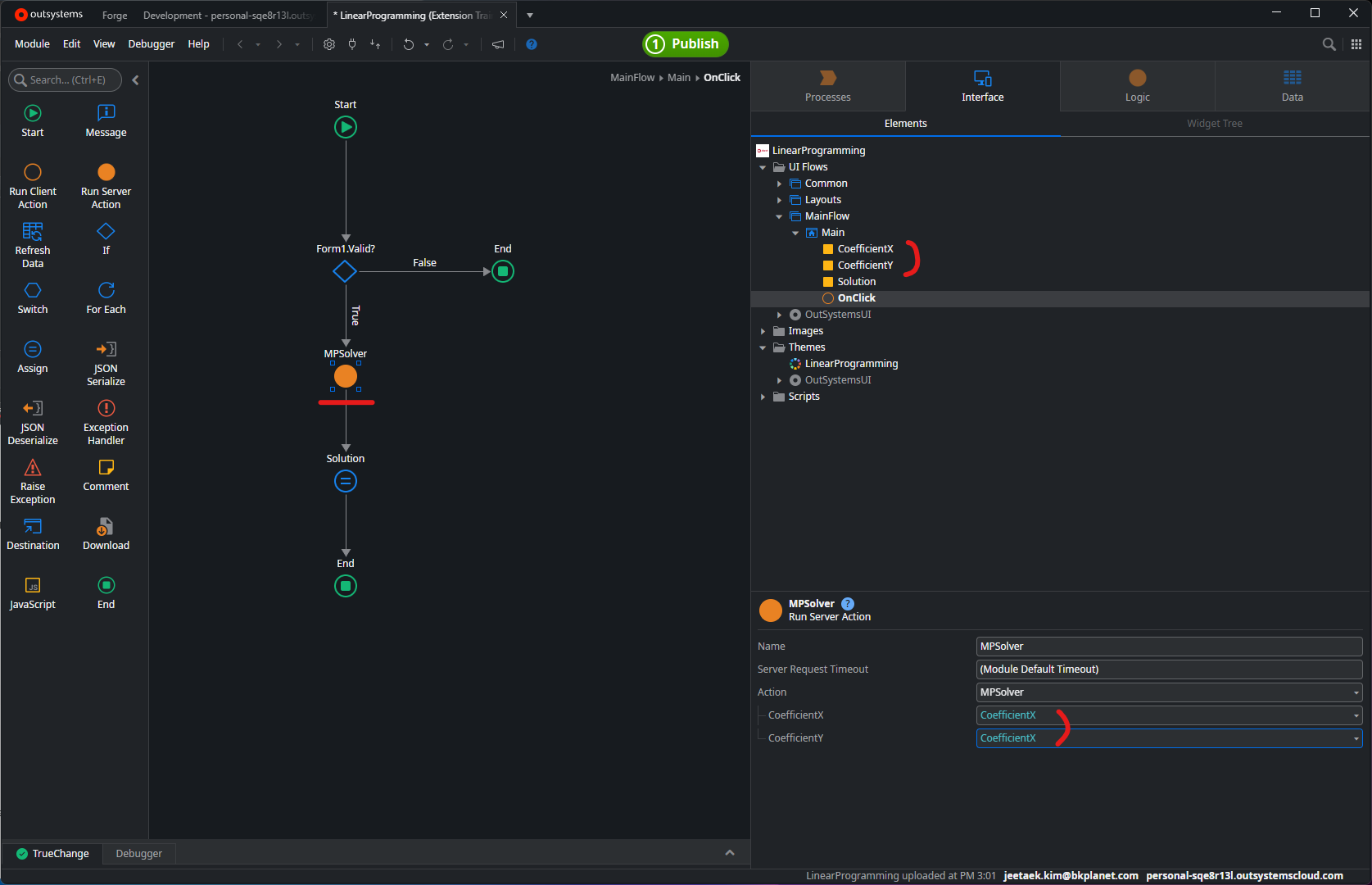
1. Ctrl+S 커맨드로 저장 후 Visual Studio를 종료하고, Integration Studio에서 Extension 모듈을 Publish 후 Integration Studio 또한 종료합니다.



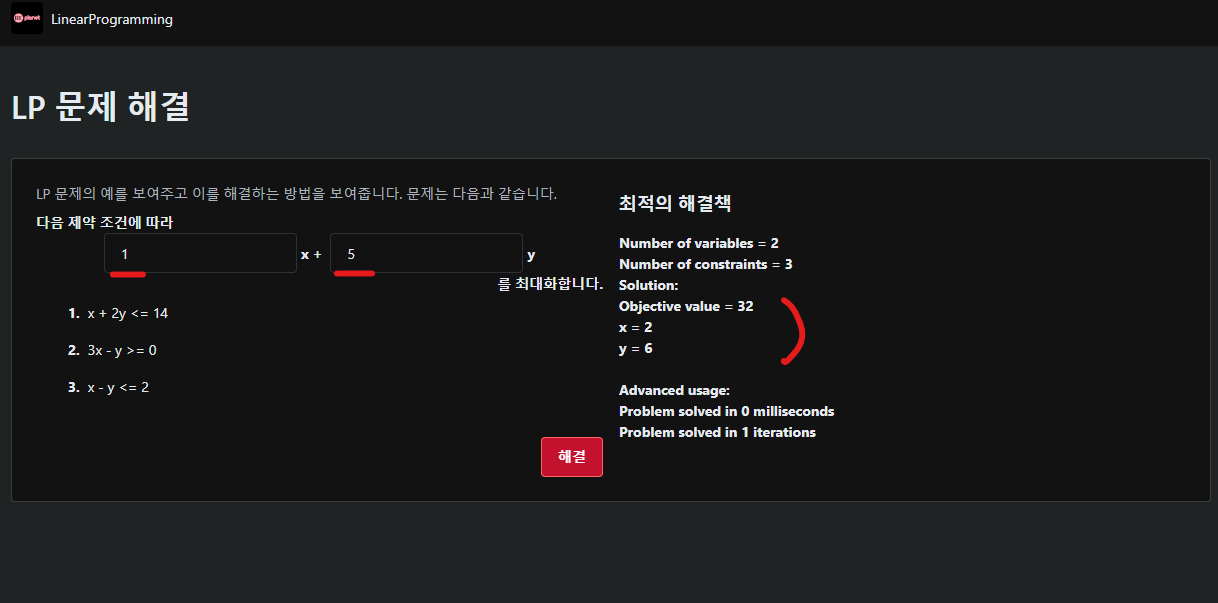
1. Service Studio에서 LinearProgramming 모듈을 열고, Dependencies를 Refresh 합니다.



1. 에러가 뜬 MPSolver 액션 노드의 인자에 Main 스크린의 지역변수들을 할당하고, 모듈을 Publish 합니다.



1. Publish가 끝나면 Open in browser 버튼을 클릭하여 브라우저로 이동하고, 최대화 문제를 수정 후 해결 버튼을 클릭하여 결과가 변경되는지 확인합니다.



# 5. 참조 링크

이 교육 매뉴얼을 통해 OutSystems의 몇 가지 필수 측면을 살펴보고 플랫폼에 대해 확인했습니다

본 주제에 대해 자세히 알아보려면 다음의 링크를 참조합니다..

1. [아웃시스템즈 개요](https://www.outsystems.com/training/courses/173/outsystems-overview/)
2. [서비스 스튜디오 개요](https://www.outsystems.com/training/courses/174/service-studio-overview/)
3. [아웃시스템 개발 소개](https://www.outsystems.com/training/courses/122/intro-to-outsystems-development/)
4. [인티그레이션 스튜디오](https://learn.outsystems.com/training/journeys/using-csharp-code-473)

No.1 Low Code Development Platform

관련 사항 질문을 위한 연락처:

Tel. 02-516-1244 / bkplanet@bkplanet.com