

Contents

Visual Studio의 XAML 도구

XAML 개요

XAML 디자이너

Visual Studio 및 Blend에서 XAML 디자인

XAML 디자이너 개요

요소 작업

개체를 레이아웃 컨테이너로 구성

리소스 만들기 및 적용

연습: 데이터에 바인딩

프로젝트 코드 디버그 또는 비활성화

Visual Studio용 Blend

개요

도형 및 패스 그리기

개체 스타일 수정

개체에 애니메이션 적용

데이터 표시

참고

XAML 오류 및 경고

XAML 디자이너에 대한 바로 가기 키

Blend의 바로 가기 키

아트보드 보조 키(Blend)

펜 도구 보조 키(Blend)

직접 선택 도구 보조 키(Blend)

XAML 디버그

디버그하는 동안 XAML 속성 검사

XAML 핫 다시 로드

XAML 코드 작성 및 실행 중인 XAML 코드 디버그

문제 해결

Blend에서 XAML 디버그

[UWP 앱 디버그 >>](#)

[WPF\(Windows Presentation Foundation\)](#)

[시작](#)

[WPF 앱 디버그](#)

[WPF 디버그](#)

[WPF 트리 시각화 도우미 사용](#)

[WPF 추적 정보 표시](#)

XAML 개요

2020-06-08 • 2 minutes to read • [Edit Online](#)

XAML(Extensible Application Markup Language)은 XML을 기반으로 하는 선언적 언어입니다. XAML은 다음과 같은 유형의 애플리케이션에서 사용자 인터페이스를 빌드하는 데 광범위하게 사용됩니다.

- [Windows Presentation Foundation \(WPF\) 앱](#)
- [UWP \(유니버설 Windows 플랫폼\) 앱](#)
- [Xamarin.Forms 앱](#)

다음 XAML 코드는 간단한 단추 컨트롤을 정의합니다.

```
<Button Click="ButtonClick">Show updates</Button>
```

XAML은 [Windows WF\(WorkFlow Foundation\) 앱](#)에서 워크플로를 정의하는 데에도 사용됩니다.

XAML 디자이너

Visual Studio 및 Blend for Visual Studio는 WPF, UWP 및 Xamarin.Forms 앱의 UI(사용자 인터페이스)를 빌드하는 데 유용한 XAML 디자이너를 제공합니다. 도구 상자 또는 자산 창에서 컨트롤을 끌어 속성 창에서 속성을 설정할 수 있습니다. 이렇게 하면 Visual Studio 및 Blend for Visual Studio 해당 XAML 코드를 만듭니다. XAML 코드를 직접 편집하려는 경우에도 이 작업을 수행할 수 있습니다.

이 설명서 집합의 문서에서는 Visual Studio 및 Blend for Visual Studio의 XAML 디자이너에 대해 설명합니다.

새로운 기능

최신 정보는 Visual studio에서 [xaml 개발자 도구의 새로운 기능 2019](#) 블로그 게시물, [visual studio 2019 버전 16.7 PREVIEW 1의 xaml 도구 향상 버전 Preview 1](#) 블로그 게시물 및 YouTube의 [Visual STUDIO 비디오](#)에 있는 새로운 xaml 기능을 참조하세요.

참고 항목

- [WPF 앱의 XAML](#)
- [UWP 앱의 XAML](#)
- [Xamarin.Forms 앱의 XAML](#)

Visual Studio 및 Blend for Visual Studio에서 XAML 디자인

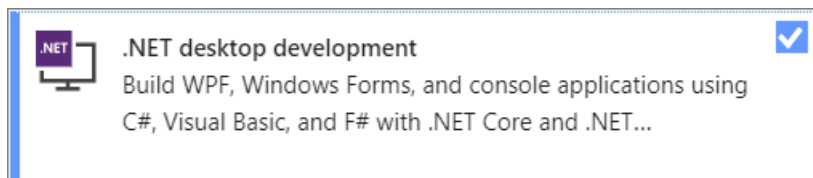
2020-05-08 • 13 minutes to read • [Edit Online](#)

Visual Studio 및 Blend for Visual Studio는 다양한 앱 유형에 XAML을 사용하여 유용한 사용자 인터페이스 및 풍부한 미디어 환경을 빌드할 수 있게 해주는 시각적 도구를 제공합니다. 두 IDE(통합 개발 환경)에서 시각적 XAML 편집기(디자이너)를 비롯한 일반적인 기능을 공유합니다. WPF 및 UWP 플랫폼을 지원하는 Blend for Visual Studio에서는 시각적 상태를 디자인하고 애니메이션을 만드는 추가 도구를 제공합니다.

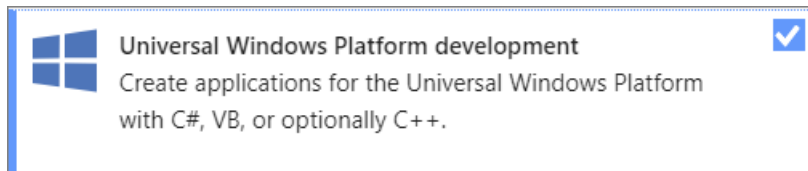
Visual Studio와 Blend for Visual Studio 간에 전환할 수 있으며, 두 IDE에서 같은 프로젝트를 동시에 열어 놓을 수도 있습니다. 한 IDE에서 XAML 파일에 저장된 변경 사항을 다른 IDE로 전환할 때 자동 다시 로드를 통해 적용할 수 있습니다. 두 IDE 중 하나에서 **도구 > 옵션 > 환경 > 문서**로 이동 하여 다시 로드 동작을 제어할 수 있습니다.

설치

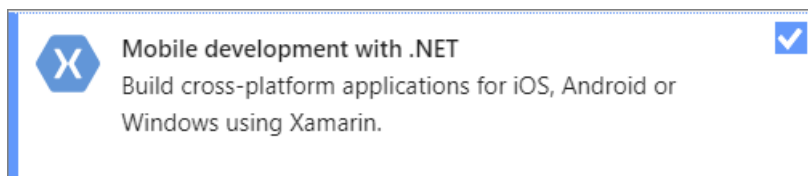
- WPF 앱을 만들려면 Visual Studio에서 **.NET 데스크톱 개발** 워크로드를 설치합니다. Blend for Visual Studio도 함께 설치됩니다.



- UWP 앱을 만들려면 Visual Studio에서 **유니버설 Windows 플랫폼 개발** 워크로드를 설치합니다. Blend for Visual Studio도 함께 설치됩니다.



- Xamarin.Forms 앱을 만들려면 Visual Studio에서 **.NET을 사용한 모바일 개발** 워크로드를 설치합니다. Blend for Visual Studio는 설치되지 *않습니다*. Blend에서는 Xamarin.Forms 앱을 지원하지 않습니다.



공유 기능

기본적인 개발 작업의 경우 대부분 Visual Studio 및 Blend for Visual Studio에서 약간의 차이점은 있지만 동일한 창과 기능 집합을 공유합니다. 중요 사항은 다음과 같습니다.

- IntelliSense:** 두 IDE는 모두 문 완성 같은 IntelliSense 기능을 지원 합니다.
- 디버깅:** 앱이 실행 되는 동안 코드에서 중단점을 설정 하여 실행 중인 응용 프로그램을 디버깅 하고, **찾다 시 로드** 를 사용 하여 XAML 코드를 변경 하는 등 **Visual Studio** 및 **Blend for Visual Studio**에서 디버그할 수 있습니다. Visual Studio와 일관된 디버깅 환경을 유지하기 위해 Blend for Visual Studio에는 Visual Studio의 디버깅 창과 도구 모음이 대부분 포함되어 있습니다.

- **파일 다시 로드:** Visual Studio 또는 Blend for Visual Studio에서 XAML 파일을 편집할 수 있습니다. 편집 후 저장된 파일은 IDE를 전환할 때 자동으로 다시 로드됩니다. 두 IDE 중 하나에서 **도구 > 옵션 > 환경 > 문서**로 이동 하여 다시 로드 동작을 제어할 수 있습니다.
- **동기화 된 레이아웃 및 설정:** 동일한 개인 설정 계정을 사용 하여 로그인 하면 Visual Studio 또는 Blend for Visual Studio에 대한 디자인 사용자 지정 도구 창 레이아웃 및 설정 기본 설정이 장치 및 버전 간에 동기화 됩니다. [여러 컴퓨터 간에 설정 동기화](#)를 참조하세요.

Blend for Visual Studio의 고급 기능

생산성을 향상시키려면 다음과 같은 작업에 Blend for Visual Studio를 사용하는 것이 좋습니다. 이러한 영역에 대해 Blend for Visual Studio는 Visual Studio 디자이너나 코드를 단독으로 사용할 때보다 더 많은 기능을 제공합니다.

작업	VISUAL STUDIO	BLEND FOR VISUAL STUDIO	추가 정보
시각적 상태 디자인	시각적 상태를 디자인하는데 도움이 되는 도구는 없습니다. 프로그래밍 방식으로 도구를 만들어야 합니다.	디자인 도구를 사용하여 해당 상태에 따라 컨트롤의 모양을 변경합니다.	시각적 상태
애니메이션 만들기	애니메이션을 만들 수 있는 디자인 도구가 없습니다. 프로그래밍 방식으로 애니메이션을 만들어야 합니다. 따라서 WPF의 애니메이션, 타이밍 시스템 및 광범위한 코딩 전문 기술을 이해하고 있어야 합니다.	애니메이션을 시각적으로 만들고 Blend for Visual Studio에서 미리 볼 수 있습니다. 코드로 애니메이션을 빌드하는 것보다 더 빠르고 정확합니다. 사용자 상호 작용을 처리하는 트리거를 추가하고 이벤트 처리기 및 기타 기능을 추가하는 코드로 전환할 수 있습니다.	개체에 애니메이션 적용
보다 쉽게 조작할 수 있도록 도형 및 텍스트를 패스로 변환	지원 안 됨	도형을 패스(더 우수한 편집 컨트롤 제공)로 변환하여 도형(예: 사각형 및 타원)에 대해 미세하거나 큰 변경을 수행할 수 있습니다. 패스 모양을 변경하거나 패스를 결합하고 여러 도형에서 복합형 패스를 만들 수 있습니다. 또한 텍스트 블록을 패스로 변환하여 벡터 이미지로 조작할 수 있습니다.	도형 및 패스 그리기

작업	VISUAL STUDIO	BLEND FOR VISUAL STUDIO	추가 정보
컨트롤, 템플릿 및 스타일 편집	코딩 및 WPF 스타일, 템플릿에 대한 지식이 필요합니다.	<p>모든 이미지를 컨트롤로 변환합니다.</p> <p>템플릿 편집 도구를 사용하여 몇 번의 마우스 클릭만으로 컨트롤, 스타일 및 템플릿을 변경할 수 있습니다.</p> <p>예를 들어 Blend for Visual Studio 스타일 리소스를 사용하여 단추, 목록 상자, 스크롤 막대, 메뉴 등과 같은 공용 WPF 컨트롤을 구현하고 Blend for Visual Studio에서 해당 색, 스타일 또는 기본 템플릿을 직접 변경할 수 있습니다. 그런 다음 원하는 경우 마무리 작업을 위해 코드로 전환할 수 있습니다.</p>	개체 스타일 수정
데이터에 UI 연결	<p>SQL Server 데이터베이스, WCF 또는 웹 서비스, 개체, SharePoint 목록 등의 리소스에서 데이터 소스를 만든 다음 UI 컨트롤에 데이터 소스를 바인딩할 수 있습니다.</p> <p>대화형 디자인 환경의 경우 디자인 타임 데이터는 직접 만들어야 합니다.</p>	<p>.NET Framework 앱의 경우 프로토타입 및 테스트를 위해 쉽게 샘플 데이터를 만듭니다. 준비가 끝나면 라이브 데이터로 전환합니다.</p> <p>Blend for Visual Studio의 데이터 생성 기능은 매우 우수한 기능으로(즉시 이름, 숫자, URL 및 사진을 쉽게 추가할 수 있음) 시간을 대폭 절약하도록 도와줍니다.</p> <p>라이브 데이터의 경우 UI 컨트롤을 XML 파일이나 모든 CLR 데이터 소스에 바인딩할 수 있습니다.</p>	데이터 표시

고급 XAML 디자인에 대한 자세한 내용은 [Blend for Visual Studio를 사용하여 UI 만들기](#)를 참조하세요.

참고 항목

- [XAML 개요](#)
- [Blend for Visual Studio 개요](#)

XAML 디자이너를 사용하여 UI 만들기

2020-05-08 • 24 minutes to read • [Edit Online](#)

Visual Studio 및 Blend for Visual Studio의 XAML 디자이너는 WPF 및 UWP와 같은 XAML 기반 응용 프로그램을 디자인 하는 데 도움이 되는 시각적 인터페이스를 제공 합니다. 도구 상자 창(Blend for Visual Studio의 자산 창)에서 컨트롤을 끌고 속성 창에서 속성을 설정하여 앱에 대한 사용자 인터페이스를 만들 수 있습니다. XAML 뷰에서 직접 XAML을 편집할 수도 있습니다.

고급 사용자의 경우 [XAML 디자이너를 사용자 지정](#)할 수 있습니다.

NOTE

Xamarin.iOS는 XAML 디자이너를 지원 하지 않습니다. 앱이 실행 되는 동안 Xamarin Forms XAML UI를 보고 편집 하려면 Xamarin.iOS에 대해 XAML 핫 다시 로드를 사용 합니다. 자세한 내용은 [xamarin.iOS에 대 한 XAML 핫 다시 로드 \(미리 보기\)](#) 페이지를 참조 하세요.

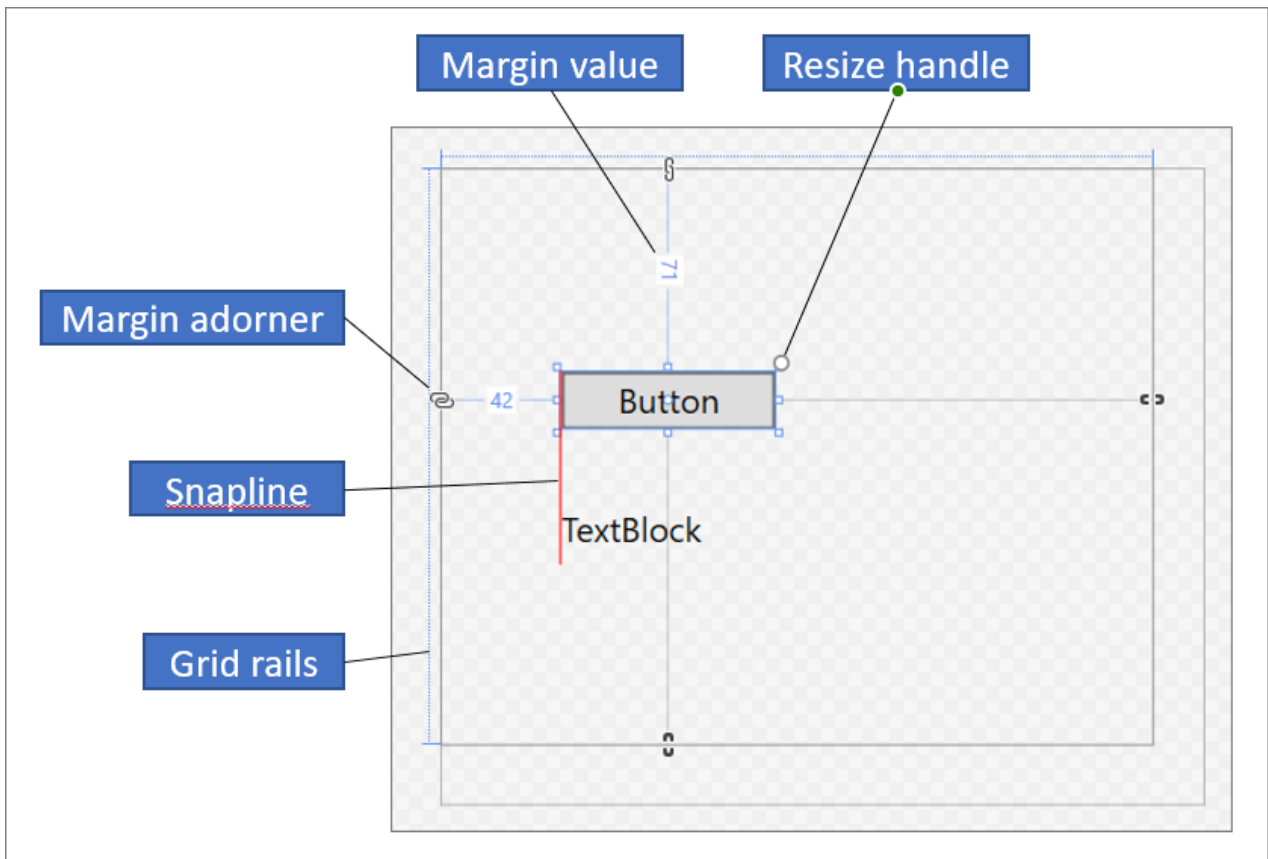
XAML 디자이너 작업 영역

XAML 디자이너의 작업 영역은 몇 가지 그래픽 인터페이스 요소로 구성됩니다. 여기에는 *아트보드*(시각적 디자인 화면), XAML 편집기, 문서 개요 창(Blend for Visual Studio의 개체 및 타임라인 창) 및 속성 창이 포함됩니다. XAML 디자이너를 열려면 **솔루션 탐색기**에서 XAML 파일을 마우스 오른쪽 단추로 클릭하고 **뷰 디자이너**를 선택합니다.

XAML 디자이너는 앱의 렌더링된 XAML 태그에 대한 XAML 뷰 및 동기화된 디자인 뷰를 제공합니다. Visual Studio 또는 Blend for Visual Studio에서 XAML 파일을 연 상태에서, **디자인** 및 **XAML** 탭을 사용하여 디자인 뷰와 XAML 뷰 사이에서 전환할 수 있습니다. **창 바꾸기** 단추 를 사용하여 아트보드 또는 XAML 편집기 중 하나의 맨 위에 나타나는 창을 전환할 수 있습니다.

디자인 보기

디자인 뷰에서 아트보드를 포함하는 창이 활성 창이고 이를 기본 작업 화면으로 사용할 수 있습니다. 요소를 추가하거나 그리거나, 수정하여 앱에서 페이지를 시각적으로 디자인할 수 있습니다. 자세한 내용은 [XAML 디자이너에서 요소 작업](#)을 참조하세요. 이 그림은 디자인 뷰에서 아트보드를 보여 줍니다.



이러한 기능은 아트보드에서 사용할 수 있습니다.

맞춤선

맞춤선은 빨간색 파선 선으로 표시되는 맞춤 경계선으로, 컨트롤의 가장자리가 맞춰진 경우나 텍스트 기준선이 맞춰진 경우를 나타냅니다. 맞춤선에 맞추기를 사용하도록 설정한 경우에만 맞춤 경계선이 나타납니다.

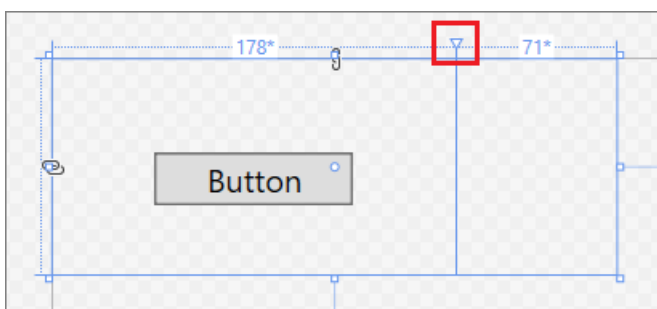
모눈 레일

모눈 레일은 Grid 패널의 행과 열을 관리하는 데 사용됩니다. 행과 열을 만들고 삭제하며, 상대적인 너비와 높이를 조정할 수 있습니다. 아트보드의 왼쪽에 나타나는 세로 모눈 레일은 행에 사용되고 맨 위에 나타나는 가로줄은 열에 사용됩니다.

모눈 표시기

모눈 표시기는 모눈 레일에 세로줄 또는 가로줄이 연결된 삼각형으로 나타납니다. 모눈 표시기를 끌면 마우스를 이동할 때 인접한 열이나 행의 너비 또는 높이가 그에 따라 업데이트됩니다.

모눈 표시기는 눈금의 행 및 열에 대한 높이 및 너비를 제어하는 데 사용됩니다. 모눈 레일에서 클릭하여 새 열 또는 행을 추가할 수 있습니다. 두 개 이상의 열 또는 행이 포함된 모눈 패널에 대한 새로운 행 또는 열 줄을 추가하는 경우 미니 도구 모음이 레일 외부에 나타나며, 이 도구 모음에서 너비와 높이를 명시적으로 설정할 수 있습니다. 미니 도구 모음을 사용하면 눈금 행 및 열에 대한 크기 조정 옵션을 설정할 수 있습니다.



크기 조정 핸들

크기 조정 핸들이 선택된 컨트롤에 나타나며 이 핸들을 사용하여 컨트롤의 크기를 조정할 수 있습니다. 컨트롤의 크기를 조정하면 너비 및 높이 값이 나타나므로 컨트롤의 크기를 조정하는 데 도움이 됩니다. **디자인** 뷰에서 컨트롤을 조작하는 방법에 대한 자세한 내용은 [XAML 디자이너에서 요소 작업](#)을 참조하세요.

여백

여백은 컨트롤 가장자리와 해당 컨테이너 가장자리 사이의 고정된 공간 크기를 나타냅니다. 속성 창의 레이아웃 아래에 있는 **여백** 속성을 사용하여 컨트롤의 여백을 설정할 수 있습니다.

여백 표시기

여백 표시기를 사용하여 레이아웃 컨테이너를 기준으로 요소의 여백을 변경합니다. 여백 표시기가 열려 있으면 여백이 설정되지 않고 여백 표시기에서 끊어진 체인을 표시합니다. 여백을 설정하지 않으면 런타임에 레이아웃 컨테이너 크기를 조정할 때 요소가 제자리에 유지됩니다. 여백 표시기가 닫힌 경우 여백 표시기는 끊어지지 않은 체인을 표시하고 런타임에 레이아웃 컨테이너의 크기가 조정될 때 요소도 여백과 함께 이동됩니다(여백은 고정됨).

요소 핸들

요소 주위의 파란색 상자 모퉁이 위로 포인터를 이동하면 아트보드에 표시되는 요소 핸들을 사용하여 요소를 수정할 수 있습니다. 이러한 핸들을 사용하면 회전, 크기 조정, 대칭 이동 또는 이동 작업을 수행하거나 요소에 모퉁이 반경을 추가할 수 있습니다. 요소 핸들의 기호는 함수별로 다르며 포인터의 정확한 위치에 따라 변경됩니다. 요소 핸들이 보이지 않으면 요소를 선택했는지 확인합니다.

디자인 뷰에서 다음과 같이 창의 왼쪽 아래 영역에서 추가 아트보드 명령을 사용할 수 있습니다.



이러한 명령은 이 도구 모음에서 사용할 수 있습니다.

확대/축소

확대/축소를 사용하면 디자인 화면의 크기를 조정할 수 있습니다. 12.5%부터 800%까지 확대/축소하거나 **선택 영역에 맞춤** 및 **모두에 맞춤**과 같은 옵션을 선택할 수 있습니다.

맞춤 모눈 숨기기

눈금선을 표시하는 맞춤 모눈을 표시하거나 숨깁니다. **모눈선에 맞추기** 또는 **맞춤선에 맞추기**를 사용할 때 눈금선이 사용됩니다.

모눈선에 맞추기 켜기/끄기

모눈선에 맞추기를 사용하는 경우 요소가 아트 보드로 끌면 가장 가까운 가로 및 세로 모눈선에 맞춰집니다.

아트보드 배경 토글

밝은 배경과 어두운 배경 간에 전환합니다.

맞춤선에 맞추기 켜기/끄기

맞춤선을 사용하면 서로를 기준으로 컨트롤을 맞춤 수 있습니다. **맞춤선에 맞추기**를 사용하는 경우 다른 컨트롤을 기준으로 컨트롤을 끌면 가장자리와 일부 컨트롤의 텍스트가 가로 또는 세로로 정렬될 때 맞춤 경계선이 나타납니다. 맞춤 경계선은 빨간색 파선으로 나타납니다.

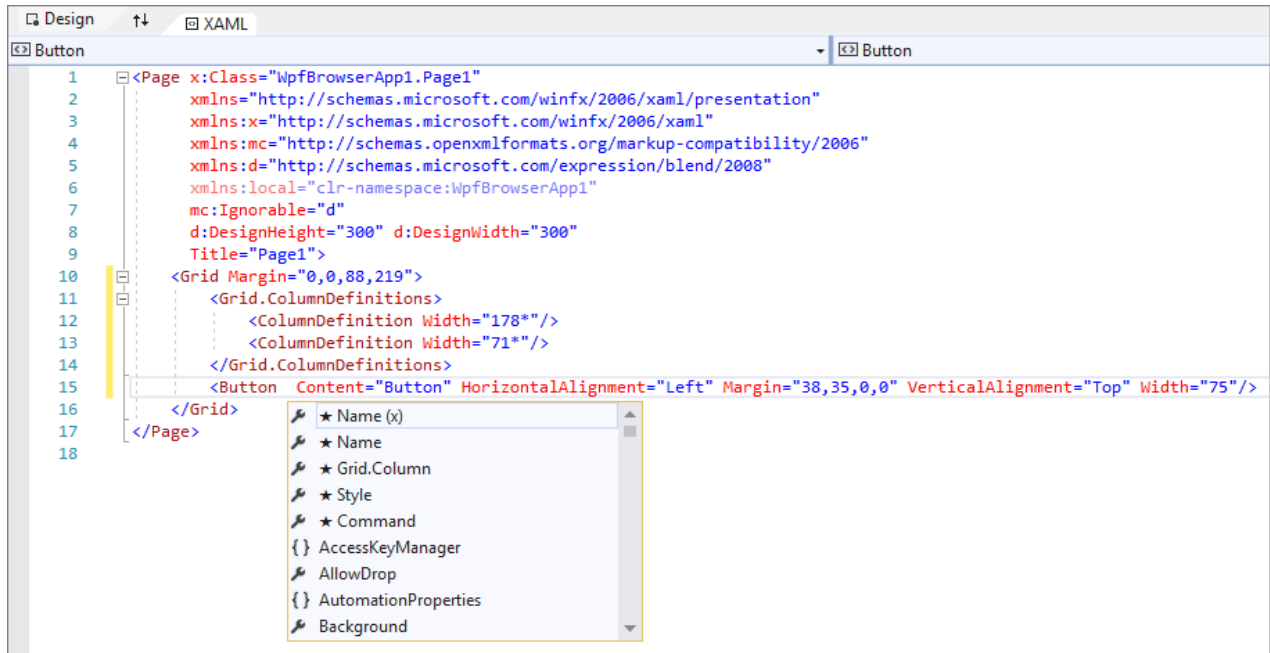
프로젝트 코드 사용 안 함

사용자 지정 컨트롤 및 값 변환기와 같은 **프로젝트 코드**를 사용하지 않도록 설정하고 디자이너를 다시 로드합니다.

XAML 뷰

Xaml 뷰에서 **xaml** 편집기가 포함된 창이 활성 창이며 xaml 편집기는 기본 제작 도구입니다. XAML(Extensible

Application Markup Language)은 애플리케이션의 사용자 인터페이스를 지정하는 데 사용할 수 있는 선언적인 XML 기반 어휘를 제공합니다. XAML 뷰에는 IntelliSense, 자동 서식 지정, 구문 강조 표시 및 태그 탐색이 포함됩니다. 다음 이미지는 IntelliSense 메뉴가 열린 XAML 뷰를 보여줍니다.

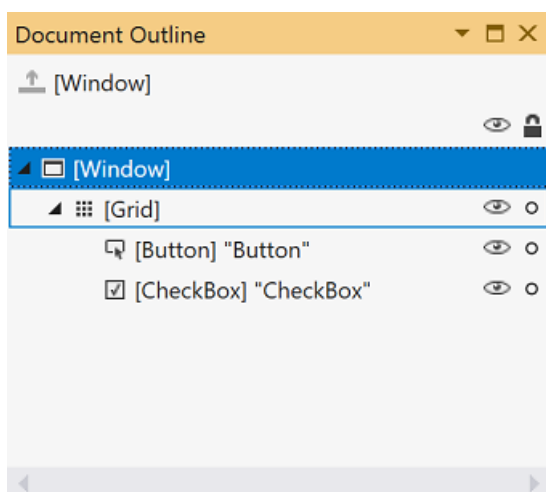


문서 개요 창

Visual Studio의 문서 개요 창은 Blend for Visual Studio의 [개체 및 타임라인](#) 창과 유사합니다. 문서 개요는 다음 작업을 수행하는 데 도움이 됩니다.

- 아트보드의 모든 요소에 대한 계층 구조를 표시합니다.
- 요소를 수정 하려면 요소를 선택 합니다. 예를 들어 계층의 주위로 이동 하거나 속성 창에서 속성을 설정 할 수 있습니다. 자세한 내용은 [XAML 디자이너에서 요소 작업](#)을 참조하세요.
- 컨트롤인 요소의 템플릿을 만들고 수정합니다.
- [애니메이션을 만듭니다](#)(Blend for Visual Studio에만 해당).

Visual Studio에서 문서 개요 창을 보려면 메뉴 모음에서 **창 > 문서 개요 보기 >** 를 선택 합니다. Blend for Visual Studio에서 개체 및 타임라인 창을 보려면 메뉴 모음에서 **문서 개요 보기 >** 를 선택 합니다.



문서 개요/개체 및 타임라인 창의 기본 보기에는 트리 구조의 문서 계층 구조가 표시됩니다. 문서 개요의 계층적 특성을 사용하여 다양한 수준의 세부 정보에서 문서를 검사하고 단독으로 또는 그룹으로 요소를 잠그고 숨길 수 있습니다. 문서 개요/개체 및 타임라인 창에서 다음 옵션을 사용할 수 있습니다.

표시/숨기기

아트보드 요소를 표시하거나 숨깁니다. 표시된 경우 눈 기호로 표시됩니다. **Ctrl+h** 를 **Shift+**눌러 요소를 숨기고 **ctrl+h** 를 눌러 표시할 수도 있습니다.

잠금/잠금해제

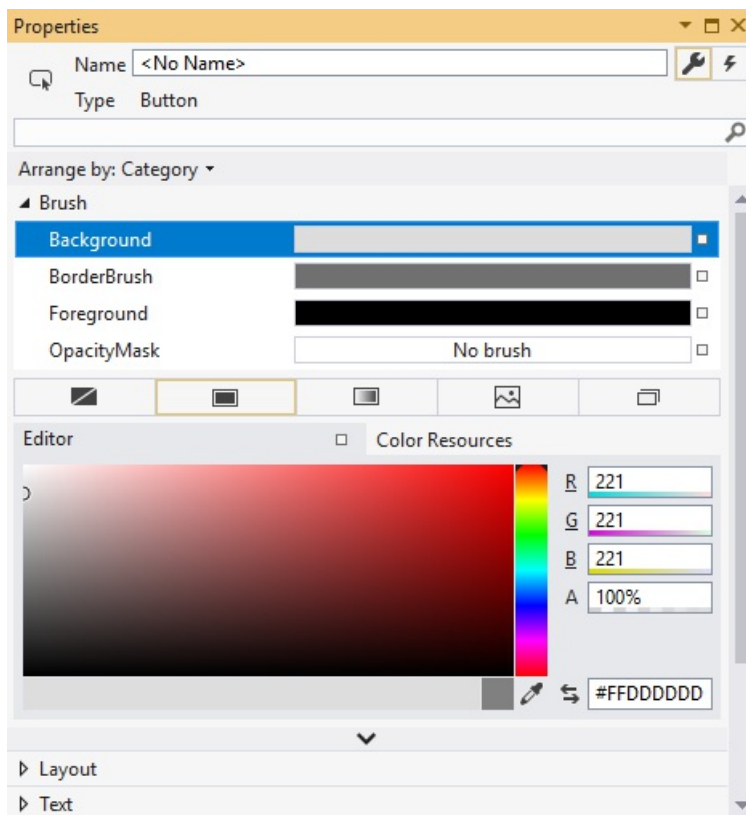
아트보드 요소를 잠그거나 잠금 해제합니다. 잠긴 요소는 수정할 수 없습니다. 잠겨 있는 경우 자물쇠 기호로 표시됩니다. **Ctrl+i** 을 눌러 요소 **Shift+**를 잠그고 **ctrl+i** 키를 눌러 잠금을 해제할 수도 있습니다.

범위를 **pageRoot**로 되돌립니다.

문서 개요/개체 및 타임라인 창의 위쪽에 있는 옵션은 위쪽 화살표 기호를 표시하며 이전 범위로 이동합니다. 범위 상향 지정은 스타일이나 템플릿의 범위에 있을 경우에만 적용할 수 있습니다.

속성 창

속성 창에서 컨트롤에 대한 속성 값을 설정할 수 있습니다. 다음과 같이 나타납니다.



속성 창의 위쪽에는 다양한 옵션이 있습니다.

- 이름 상자에서 현재 선택된 요소의 이름을 변경합니다.
- 왼쪽 위 모서리에 현재 선택한 요소를 나타내는 아이콘이 있습니다.
- 속성을 범주별로 또는 사전순으로 정렬하려면 정렬 기준목록에서 범주, 이름 또는 소스 를 클릭합니다.
- 컨트롤에 대한 이벤트의 목록을 보려면 번개 기호를 표시하는 이벤트 단추를 클릭합니다.
- 속성을 검색하려면 검색 상자에 속성 이름을 입력하기 시작합니다. 사용자가 입력할 때 검색 조건과 일치 하는 속성이 속성 창에 표시 됩니다.

일부 속성에서는 아래쪽 화살표 단추를 선택하여 고급 속성을 설정할 수 있습니다.

각 속성 값의 오른쪽에 상자 기호로 나타나는 속성 표식이 있습니다. 속성 표식의 모양은 속성에 적용되는 리소스 또는 데이터 바인딩이 있는지 여부를 나타냅니다. 예를 들어 흰색 상자 기호는 기본값을 나타내고, 검은색 상자 기호는 일반적으로 로컬 리소스가 적용되었음을 나타내고, 주황색 상자는 일반적으로 데이터 바인딩이 적용되었음을 나타냅니다. 속성 표식을 클릭하면 스타일의 정의로 이동하거나, 데이터 바인딩 작성기를 열거나, 리소스 선택기를 엽니다.

속성 사용 및 이벤트 처리에 대한 자세한 내용은 [컨트롤 및 패턴 소개](#)를 참조하세요.

참고 항목

- [XAML 디자이너의 요소 작업](#)
- [방법: 리소스 만들기 및 적용](#)
- [연습: XAML 디자이너에서 데이터에 바인딩](#)

XAML 디자이너의 요소 작업

2020-05-08 • 19 minutes to read • [Edit Online](#)

컨트롤, 레이아웃 및 모양 등의 요소를 XAML 앱에 코드로 추가하거나 XAML 디자이너를 사용하여 추가할 수 있습니다. 이 항목에서는 Visual Studio 또는 Blend for Visual Studio의 XAML 디자이너에서 요소에 대해 작업하는 방법을 설명합니다.

레이아웃에 요소 추가

*레이아웃*은 UI에서 요소의 크기를 조정하고 배치하는 프로세스입니다. 시각적 요소를 배치하려면 **패널** 레이아웃에 가져다 놓아야 합니다. `Panel`에는 `FrameworkElement` 형식의 컬렉션인 자식 속성이 있습니다. `Canvas`, `StackPanel` 및 `Panel` `Grid`와 같은 다양한 자식 요소를 사용하여 레이아웃 컨테이너 역할을 하고 페이지에서 요소를 배치 및 정렬할 수 있습니다.

기본적으로는 `Grid` 패널은 페이지 또는 폼 내에서 최상위 레이아웃 컨테이너로 사용됩니다. 최상위 페이지 레이아웃 내에서 레이아웃 패널, 컨트롤 또는 다른 요소를 추가할 수 있습니다.

XAML 디자이너의 레이아웃에 요소를 추가하려면 다음 중 하나를 수행하세요.

- 도구 상자에서 요소를 두 번 클릭하거나 도구 상자에서 요소를 선택하고 **Enter** 키를 누릅니다.
- 요소를 도구 상자에서 아트보드로 끌어 놓습니다.
- 도구 상자에서 그리기 도구(예: [타원](#) 또는 [사각형](#)) 중 하나를 선택한 다음 활성 패널에서 요소를 그립니다.

요소의 쌓기 순서 변경

XAML 디자이너의 아트보드에 두 요소가 있는 경우 한 요소가 쌓기 순서 대로 다른 요소 앞에 표시됩니다. 문서 개요 창의 요소 목록 아래쪽에는 맨 앞의 요소가 있습니다(요소에 대해 `ZIndex` 속성이 설정된 경우 제외). 페이지, 폼 또는 레이아웃 컨테이너에 요소를 삽입할 때 요소가 활성 컨테이너 요소의 다른 요소 앞에 자동으로 배치됩니다. 요소의 순서를 변경하려면 **Order** 명령을 사용하거나 [문서 개요] 창의 개체 트리에서 요소를 끌면 됩니다.

쌓기 순서를 변경하려면 다음 중 하나를 수행합니다.

- 문서 개요 창에서 요소를 위쪽이나 아래쪽으로 끌어 원하는 쌓기 순서를 만듭니다.
- [문서 개요] 창 또는 쌓기 순서를 변경하려는 아트보드에서 요소를 마우스 오른쪽 단추로 클릭하고, **Order**를 가리키고, 다음 중 하나를 클릭합니다.
 - 맨 앞으로 가져오기 - 요소를 순서의 맨 앞으로 가져옵니다.
 - 앞으로 가져오기 - 요소를 순서대로 한 수준 앞으로 가져옵니다.
 - 뒤로 보내기 - 요소를 순서대로 한 수준 뒤로 보냅니다.
 - 맨 뒤로 보내기 - 요소를 순서의 맨 뒤로 보냅니다.
- [속성] 창의 레이아웃 섹션에서 `ZIndex` 속성을 변경합니다. 겹치는 요소의 경우 `ZIndex` 속성이 [문서 개요] 창에 표시되는 요소의 순서보다 우선합니다. 요소가 겹치는 경우 `Z` 인덱스 값이 더 큰 요소가 앞에 표시됩니다.

요소의 맞춤 변경

메뉴 명령을 사용하거나 맞춤선에 요소를 끌어 아트 보드에서 요소를 맞춤 수 있습니다.

맞춤선은 앱의 다른 요소를 기준으로 요소를 맞추는 데 도움이 되는 시각적 표시입니다.

메뉴 명령을 사용하여 둘 이상의 요소를 맞추려면

1. 맞춤 요소를 선택합니다. 요소를 선택할 때 **Ctrl** 키를 누르고 선택하면 두 개 이상의 요소를 선택할 수 있습니다.
2. [속성] 창의 **레이아웃** 섹션에 있는 **HorizontalAlignment** 아래에서 **Left**, **Center**, **Right** 또는 **Stretch** 속성 중 하나를 선택합니다.
3. [속성] 창의 **레이아웃** 섹션에 있는 **VerticalAlignment** 아래에서 **Top**, **Center**, **Bottom** 또는 **Stretch** 속성 중 하나를 선택합니다.

맞춤선을 사용하여 둘 이상의 요소를 맞추려면 XAML 디자이너의 둘 이상의 요소가 포함된 레이아웃에서 가장자리가 다른 요소에 맞춰지도록 요소 중 하나를 끌거나 크기를 조정합니다.

가장자리가 맞춰지는 경우 **맞춤 경계선**이 표시되어 맞춤을 나타냅니다. 맞춤 경계선은 빨간색 파선입니다. **맞춤선에 맞추기**를 사용하도록 설정한 경우에만 맞춤 경계선이 나타납니다. 맞춤 경계선을 표시하는 아트보드의 그림은 [XAML 디자이너를 사용하여 UI 만들기](#)를 참조하세요.

요소의 여백 변경

XAML 디자이너에서 여백은 아트보드의 요소 주위에 있는 빈 공간의 크기를 결정합니다. 예를 들어 여백은 요소의 바깥쪽 가장자리와 해당 요소가 들어 있는 **Grid** 패널의 경계선 사이에 있는 공간의 크기를 지정합니다. 또한 **StackPanel**에 포함된 요소 사이에 있는 공간의 크기를 지정합니다.

속성 창에서 요소의 여백을 변경하려면

1. 여백을 변경할 요소를 선택합니다.
2. [속성] 창의 **레이아웃** 아래에서 **Margin** 속성(**Top**, **Left**, **Right** 또는 **Bottom**) 값(픽셀 단위 또는 디바이스 와 독립적인 단위, 대략 1/96인치)을 변경합니다.

아트보드에서 요소의 레이아웃 컨테이너를 기준으로 요소의 여백을 변경하려면 요소를 선택하거나 요소가 레이아웃 컨테이너에 있을 때 요소 주위에 나타나는 **여백 표시기**를 클릭합니다. 여백 표시기를 표시하는 그림은 [XAML 디자이너를 사용하여 UI 만들기](#)를 참조하세요.

여백 표시기가 세로 또는 가로로 열려 있으면 해당 여백이 설정되지 않은 것이며, 여백 표시기가 닫혀 있으면 여백이 설정된 것입니다.

여백 표시기를 열고 반대쪽 여백이 설정되어 있지 않으면 반대쪽 여백이 아트 보드의 요소 위치에 따라 올바른 값으로 설정됩니다. **Left** 및 **Right** 여백과 같은 반대쪽 여백에는 항상 하나 이상의 속성이 설정됩니다.

IMPORTANT

Canvas와 같은 레이아웃 컨테이너 안에 배치된 요소에는 여백 표시기가 없습니다. **StackPanel** 안에 배치된 요소에는 **StackPanel**의 방향에 따라 왼쪽 및 오른쪽 여백이나 위쪽 및 아래쪽 여백에 대한 여백 표시기가 있습니다.

요소 그룹화 및 그룹 해제

XAML 디자이너에서 둘 이상의 요소를 그룹화하면 새로운 레이아웃 컨테이너가 만들어지고 이 컨테이너에 해당 요소가 배치됩니다. 레이아웃 컨테이너에 둘 이상의 요소를 함께 배치하면 해당 그룹의 요소가 한 요소인 것처럼 쉽게 그룹을 선택하고 이동하며 변형할 수 있습니다. 그룹화는 탐색 요소를 구성하는 단추와 같이 특정 방법으로 서로 관련된 요소를 식별하는 데 유용합니다. 요소의 그룹을 해제할 때는 요소가 들어 있는 레이아웃 컨테이너만 삭제하면 됩니다.

새 레이아웃 컨테이너에 요소를 그룹화하려면

1. 그룹화할 요소를 선택합니다. 여러 요소를 선택 하려면 **ctrl** 키를 누른 상태에서 클릭 합니다.
2. 선택한 요소를 마우스 오른쪽 단추로 클릭하고, **그룹으로 묶기**를 가리킨 다음, 그룹을 배치할 레이아웃 컨테이너의 형식을 클릭합니다.

TIP

Viewbox, **Border** 또는 **ScrollView**를 선택하여 요소를 그룹화하는 경우 요소는 **Viewbox**, **Border** 또는 **ScrollView** 내의 새 **Grid** 패널에 배치됩니다. 이러한 레이아웃 컨테이너 중 하나에서 요소의 그룹을 해제하면 **Viewbox**, **Border** 또는 **ScrollView**만 삭제되고 **Grid** 패널은 그대로 유지됩니다. **Grid** 패널을 삭제하려면 요소의 그룹을 다시 해제 합니다.

요소를 그룹 해제하고 레이아웃을 삭제하려면 그룹 해제하려는 그룹을 마우스 오른쪽 단추로 클릭하고 **그룹 해제**를 클릭합니다. [문서 개요] 창에서 선택한 항목을 마우스 오른쪽 단추로 클릭하고 **그룹으로 묶기** 또는 **그룹 해제**를 클릭하여 요소를 그룹화하거나 그룹 해제할 수도 있습니다.

요소 레이아웃 재설정

레이아웃 다시 설정 명령을 사용하여 요소의 특정 레이아웃 속성에 대한 기본값을 복원할 수 있습니다. 이 명령을 사용하여 요소의 여백, 맞춤, 너비, 높이 및 크기를 개별적으로 또는 전체적으로 다시 설정할 수 있습니다.

요소 레이아웃을 다시 설정 하려면 문서 개요 창이 나 아트 보드에서 요소를 마우스 오른쪽 단추로 클릭 한 다음 레이아웃 > 다시 설정 *PropertyName*을 선택 합니다. 여기서 *PropertyName*은 다시 설정할 속성입니다. 또는 레이아웃 > 모두 다시 설정을 선택 하 여 요소의 모든 레이아웃 속성을 다시 설정 합니다.

참고 항목

- [XAML 디자이너를 사용하여 UI 만들기](#)

XAML 디자이너에서 개체를 레이아웃 컨테이너로 구성

2020-05-08 • 9 minutes to read • [Edit Online](#)

이 문서에서는 XAML 디자이너에 대한 레이아웃 패널과 컨트롤을 설명합니다.

페이지에 개체를 표시할 위치를 가정해 보겠습니다. —개체로는 이미지, 단추, 비디오 등을 들 수 있습니다. 개체를 행 및 열로 표시하거나 가로나 세로로 한 줄로 표시하거나 고정된 위치에 표시할 수 있습니다.

페이지에 어떻게 표시할지 결정했으면 레이아웃 패널을 선택합니다. 개체를 추가할 대상이 필요하기 때문에 모든 페이지 작업은 여기에서 시작됩니다. 기본적으로 **Grid**가 설정되지만 변경할 수 있습니다.

레이아웃 패널은 페이지에 개체를 정렬하는 데 도움이 될 뿐만 아니라 그 이상의 역할을 합니다. 서로 다른 화면 크기 및 해상도를 설계하는 데에도 도움이 됩니다. 사용자가 앱을 실행할 때 레이아웃 패널의 모든 항목은 디바이스의 실제 화면 공간에 맞게 크기가 조정됩니다. 자동으로 크기가 조정되지 않도록 하려면 레이아웃 일부 또는 전체 레이아웃에 대해 이러한 동작을 재정의할 수 있습니다. 이러한 동작은 높이 및 너비 속성을 사용하여 제어할 수 있습니다.

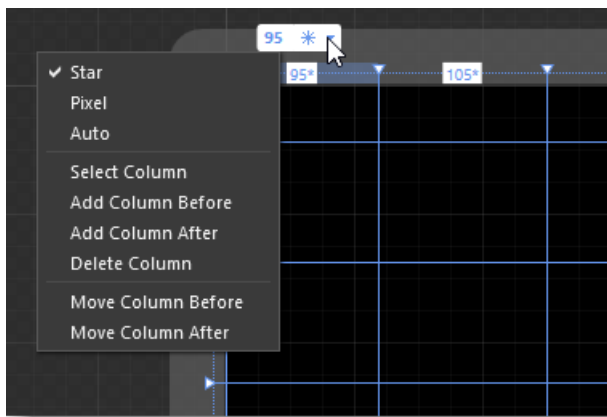
레이아웃 패널

다음 레이아웃 패널 중 하나를 선택하여 페이지를 시작합니다. 페이지에 둘 이상의 레이아웃 패널을 지정할 수 있습니다. 예를 들어 **Grid** 레이아웃 패널에서 시작한 후 **Grid**의 영역에 **StackPanel**을 추가하여 해당 요소에서 컨트롤을 세로로 정렬할 수 있습니다.

다음은 가장 많이 사용되는 레이아웃 패널이며, 다른 레이아웃 패널도 있습니다. 이 모든 항목은 Visual Studio의 도구 상자 또는 Blend for Visual Studio의 자산 패널에서 확인할 수 있습니다.

표

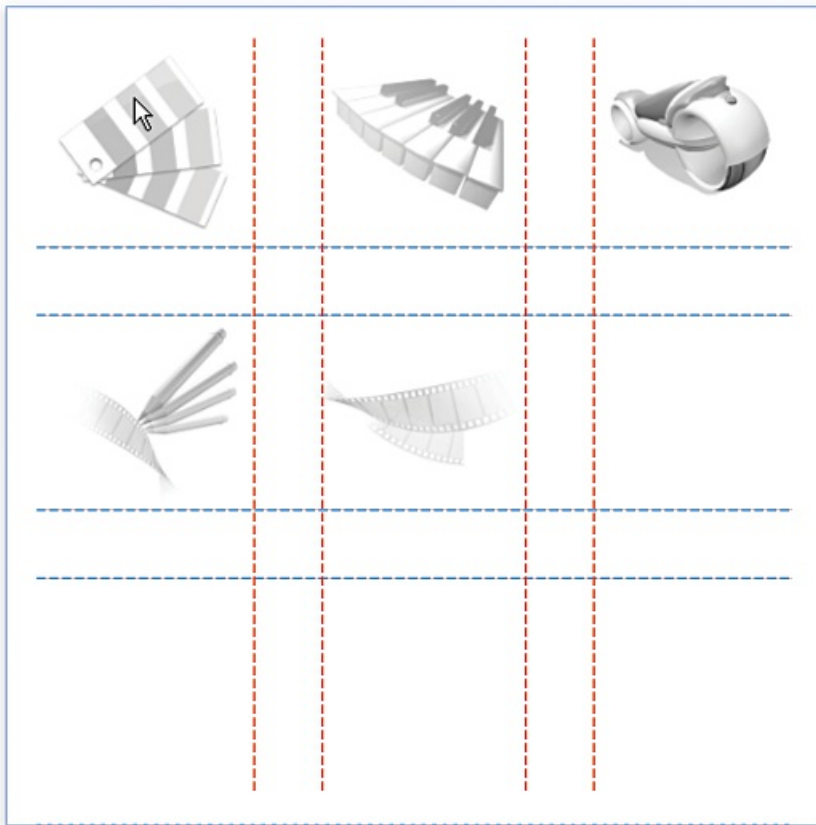
개체를 행 및 열로 정렬합니다.



UniformGrid

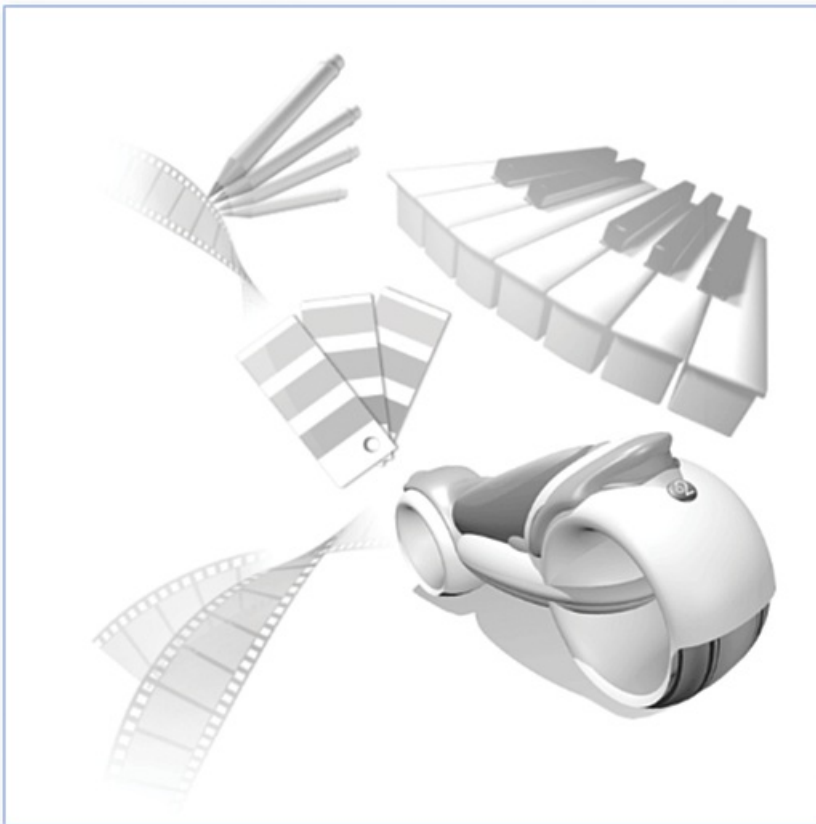
개체를 동일하게 또는 균일하게 모눈 영역에 정렬합니다. 이 패널은 이미지의 목록을 정렬할 때 매우 유용합니다.

(WPF 프로젝트에만 사용 가능)



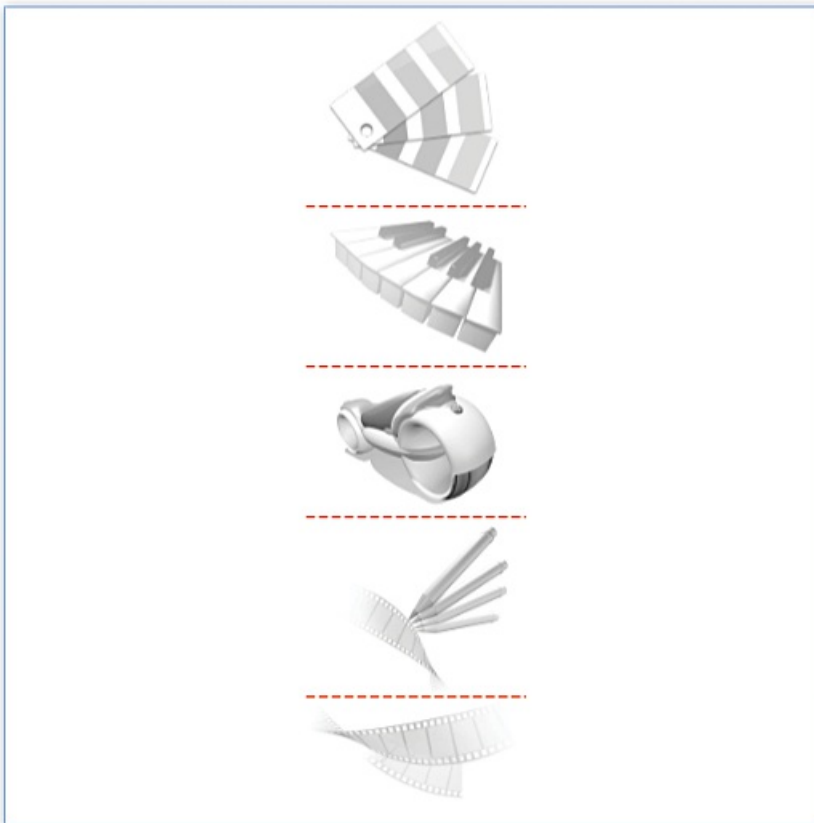
Canvas

원하는 방식으로 개체를 정렬합니다. 사용자가 앱을 실행할 때 이러한 요소는 화면에서 고정 위치에 표시됩니다.



StackPanel

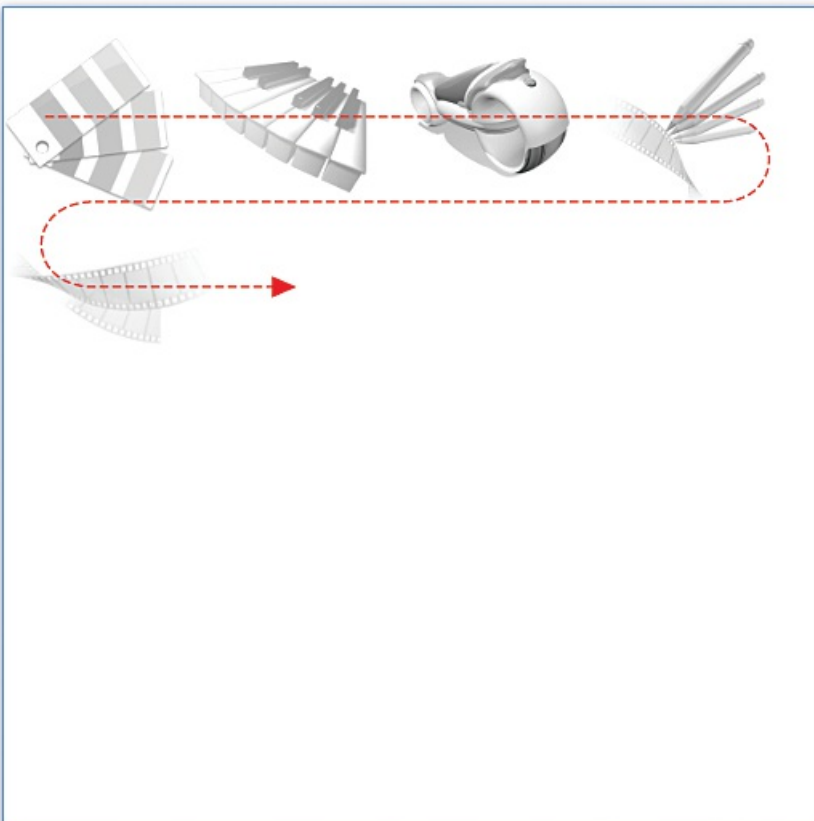
개체를 가로 또는 세로로 한 줄로 정렬합니다.



WrapPanel

개체를 순서대로 왼쪽에서 오른쪽으로 정렬합니다. 패널의 맨 오른쪽 가장자리에 공간이 부족한 경우 콘텐츠를 다음 줄로 *줄바꿈*하며 왼쪽에서 오른쪽, 위쪽에서 아래쪽 가로로 배치됩니다. 위쪽에서 아래쪽, 왼쪽에서 오른쪽 세로로 배치할 수도 있습니다.

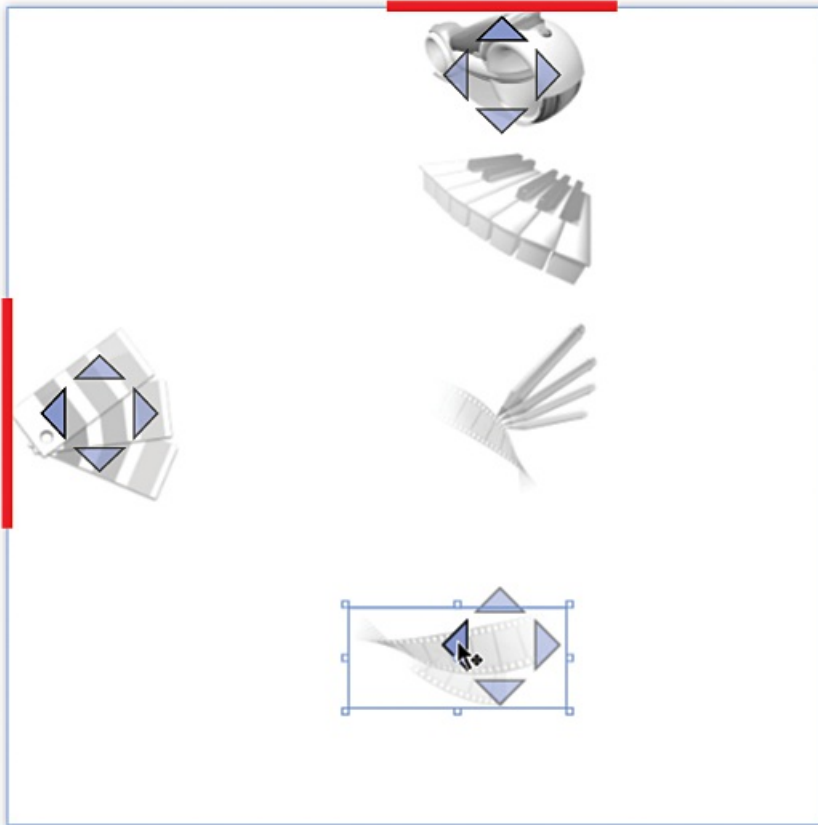
(WPF 프로젝트에만 사용 가능)



DockPanel

개체가 패널의 한쪽 가장자리에 고정되도록 개체를 정렬합니다.

(WPF 프로젝트에만 사용 가능)



짧은 비디오 보기: [▶ WPF - DockPanel](#)

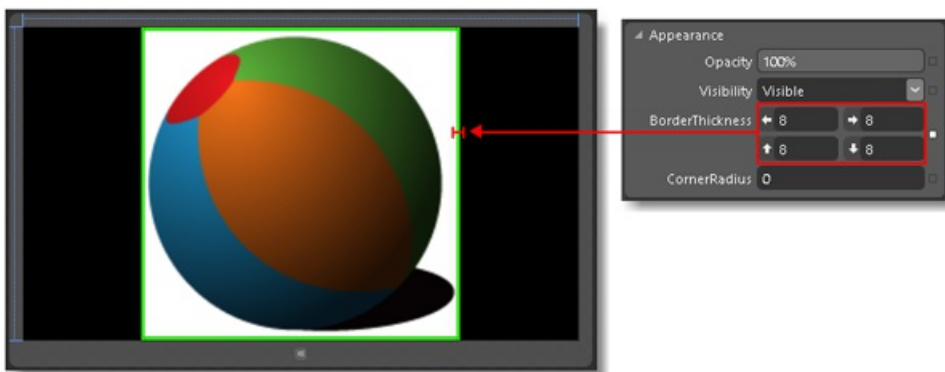
레이아웃 컨트롤

개체를 레이아웃 컨트롤에도 추가할 수 있습니다. 이러한 개체는 레이아웃 패널만큼 기능이 다양하지는 않지만 특정 시나리오에서 유용할 수 있습니다.

다음은 가장 많이 사용되는 레이아웃 컨트롤이며, 다른 레이아웃 컨트롤도 있습니다. 이 모든 항목은 Visual Studio의 도구 상자 또는 Blend for Visual Studio의 자산 패널에서 확인할 수 있습니다.

테두리

개체 주위에 테두리, 배경 또는 둘 다를 그립니다. **Border**에는 개체를 하나만 추가할 수 있습니다. 둘 이상의 개체에 테두리나 배경을 적용하려면 레이아웃 패널을 **Border**에 추가합니다. 그런 다음 해당 패널 또는 컨트롤에 개체를 추가합니다.

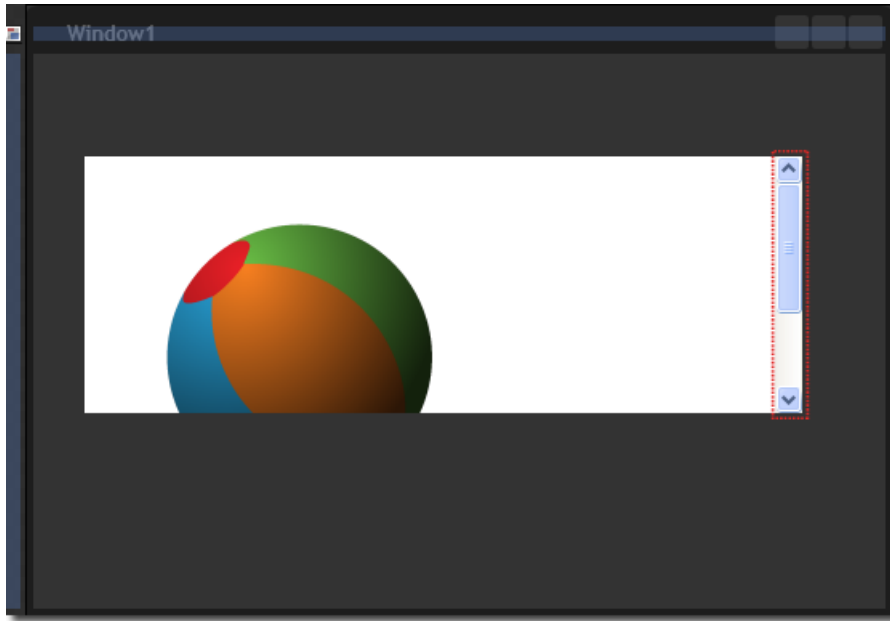


Popup

창의 사용자에게 정보나 옵션을 표시합니다. **Popup**에 개체를 하나만 추가할 수 있습니다. 기본적으로 **Popup**에는 **Grid**가 포함되지만 변경할 수 있습니다.

ScrollView

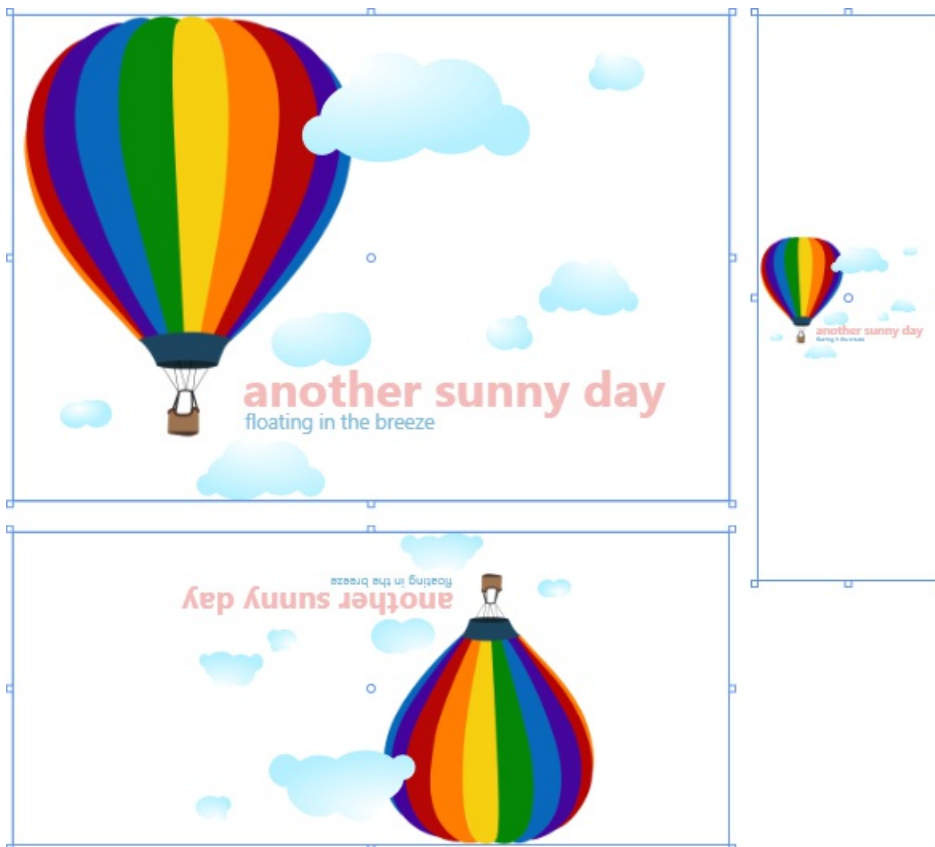
사용자가 페이지나 페이지의 특정 영역을 스크롤할 수 있습니다. **ScrollView**에 개체를 하나만 추가할 수 있으므로 **Grid** 또는 **StackPanel** 등과 같은 레이아웃 패널을 추가합니다.



Viewbox

확대/축소 컨트롤과 유사하게 개체의 비율 크기를 조정합니다. **Viewbox**에 하나의 개체만 추가할 수 있습니다. 둘 이상의 개체에 해당 효과를 적용하려면 레이아웃 패널을 **ViewBox**에 추가한 후 컨트롤을 해당 레이아웃 패널에 추가합니다.

(WPF 프로젝트에만 사용 가능)



참고 항목

- [XAML 디자이너의 요소 작업](#)

- XAML 디자이너를 사용하여 UI 만들기

방법: 리소스 만들기 및 적용

2020-05-08 • 9 minutes to read • [Edit Online](#)

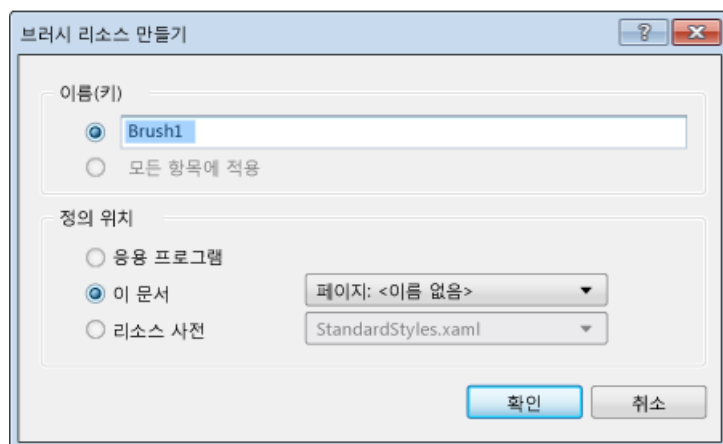
XAML 디자이너의 요소에 대한 스타일 및 템플릿은 리소스라는 다시 사용 가능한 엔터티에 저장됩니다. 스타일을 사용하면 요소 속성을 설정하고 여러 요소에 걸쳐 일관된 모양을 위해 이러한 설정을 통해 다시 사용할 수 있습니다. [ControlTemplate](#)은 컨트롤의 모양을 정의하며 리소스로 적용될 수도 있습니다. 자세한 내용은 [XAML 스타일 및 컨트롤 템플릿](#)을 참조하세요.

기존 속성인 [스타일](#) 또는 [ControlTemplate](#)으로 새 리소스를 만들 때마다 [리소스 만들기](#) 대화 상자를 사용하여 애플리케이션 수준, 문서 수준 또는 요소 수준에서 리소스를 정의할 수 있습니다. 이러한 수준에 따라 리소스를 사용할 수 있는 위치가 결정됩니다. 예를 들어 요소 수준에서 리소스를 정의하는 경우 리소스 만든 요소에만 적용할 수 있습니다. 또한 다른 프로젝트에서 다시 사용할 수 있는 별도 파일인 [리소스 사전](#)에 리소스를 저장할 수 있습니다.

새 리소스 만들기

1. XAML 디자이너에서 XAML 파일을 연 상태에서, 요소를 만들거나 문서 개요 창에서 요소를 선택합니다.
2. 속성 창에서 속성 값의 오른쪽에 있는 상자 기호로 나타나는 속성 표식을 선택한 다음 [새 리소스로 변환](#)을 선택합니다. 흰색 상자 기호는 기본값을 나타내고 검은색 상자 기호는 일반적으로 로컬 리소스가 적용된 것을 나타냅니다.

리소스를 만들 수 있는 적절한 대화 상자가 나타납니다. 이 대화 상자는 브러시에서 리소스를 만들 때 나타납니다.



3. **이름(키)** 상자에 키 이름을 입력합니다. 이 이름은 다른 요소가 리소스를 참조하도록 할 때 사용할 수 있습니다.
4. **정의 위치**에서 리소스를 정의할 위치를 지정하는 옵션을 선택합니다.
 - 애플리케이션의 모든 문서에 리소스를 사용할 수 있도록 하려면 **애플리케이션**을 선택합니다.
 - 현재 문서에서만 리소스를 사용할 수 있도록 하려면 **이 문서**를 선택합니다.
 - 리소스를 만든 요소 또는 그 자식 요소에만 리소스를 사용할 수 있도록 하려면 요소에만 **이 문서**를 선택하고 드롭다운 목록에서 **요소: 이름**을 선택합니다.
 - 다른 프로젝트에서 다시 사용할 수 있는 [리소스 사전](#) 파일에서 리소스를 정의하려면 **리소스 사전**을 클릭합니다. 그런 다음 드롭다운 목록에서 **StandardStyles.xaml**과 같은 기존 리소스 사전 파일을 선택합니다.
5. **확인** 단추를 선택하여 리소스를 만들고 리소스를 만든 요소에 적용합니다.

요소 또는 속성에 리소스 적용

1. 문서 개요 창에서 리소스를 적용하려는 요소를 선택합니다.

2. 다음 중 하나를 수행합니다.

- 속성에 리소스를 적용합니다. 속성 창에서 속성 값 옆에 있는 속성 표식을 선택하고, 로컬 리소스 또는 시스템 리소스를 선택한 다음, 표시되는 목록에서 사용 가능한 리소스를 선택합니다.

보려는 리소스가 표시되지 않으면 리소스 종류가 속성 유형과 맞지 않기 때문일 수 있습니다.

- 컨트롤에 스타일 또는 컨트롤 템플릿 리소스를 적용합니다. [문서 개요] 창에서 컨트롤의 오른쪽 클릭 메뉴(상황에 맞는 메뉴)를 열고, **템플릿 편집** 또는 **추가 템플릿 편집**을 선택하고, 리소스 적용을 선택한 다음, 표시되는 목록에서 컨트롤 템플릿의 이름을 선택합니다.

NOTE

템플릿 편집은 컨트롤 템플릿을 적용합니다. 추가 템플릿 편집은 다른 템플릿 형식을 적용합니다.

호환되는 모든 위치에 리소스를 적용할 수 있습니다. 예를 들어 **TextBox** 컨트롤의 **Foreground** 속성에 브러시 리소스를 적용할 수 있습니다.

리소스 편집

1. 아트보드 또는 문서 개요 창에서 요소를 선택합니다.

2. 속성 창에서 속성의 오른쪽에 있는 기본 또는 로컬 속성 표식을 선택하고 **리소스 편집**을 선택하여 리소스 편집 대화 상자를 엽니다.

3. 리소스에 대한 옵션을 수정합니다.

참고 항목

- [XAML 디자이너를 사용하여 UI 만들기](#)

연습: XAML 디자이너에서 데이터에 바인딩

2020-05-08 • 6 minutes to read • [Edit Online](#)

XAML 디자이너에서 아트보드와 속성 창을 사용하여 데이터 바인딩 속성을 설정할 수 있습니다. 이 연습의 예제에서는 데이터를 컨트롤에 바인딩하는 방법을 보여줍니다. 특히 이 연습에서는 `ItemCount` 라는 `DependencyProperty`가 포함된 간단한 쇼핑 카트 클래스를 만든 다음, `TextBlock` 컨트롤의 `Text` 속성에 `ItemCount` 속성을 바인딩하는 방법을 보여 줍니다.

데이터 소스로 사용할 클래스를 만들려면

1. 파일 메뉴에서 새로 만들기 > 프로젝트를 선택합니다.
2. 새 프로젝트 대화 상자에서 **Visual C#** 또는 **Visual Basic** 노드를 선택하고, **Windows** 바탕 화면 노드를 펼친 다음, **WPF 애플리케이션** 템플릿을 선택합니다.
3. **BindingTest** 프로젝트 이름을 지정한 다음 **확인** 단추를 선택합니다.
4. **MainWindow.xaml.cs**(또는 **MainWindow.xaml.vb**) 파일을 열고 다음 코드를 추가합니다. C#에서는 `BindingTest` 네임스페이스에 코드를 추가합니다(파일의 마지막 닫는 괄호 앞에 추가). Visual Basic에서는 새 클래스를 추가합니다.

```
public class ShoppingCart : DependencyObject
{
    public int ItemCount
    {
        get { return (int)GetValue(ItemCountProperty); }
        set { SetValue(ItemCountProperty, value); }
    }

    public static readonly DependencyProperty ItemCountProperty =
        DependencyProperty.Register("ItemCount", typeof(int),
            typeof(ShoppingCart), new PropertyMetadata(0));
}
```

```
Public Class ShoppingCart
    Inherits DependencyObject

    Public Shared ReadOnly ItemCountProperty As DependencyProperty = DependencyProperty.Register(
        "ItemCount", GetType(Integer), GetType(ShoppingCart), New PropertyMetadata(0))
    Public Property ItemCount As Integer
    Get
        ItemCount = CType(GetValue(ItemCountProperty), Integer)
    End Get
    Set(value As Integer)
        SetValue(ItemCountProperty, value)
    End Set
End Property
End Class
```

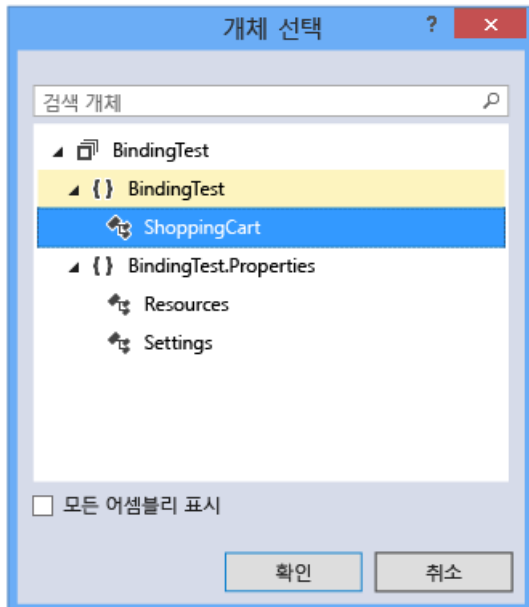
이 코드는 `PropertyMetadata` 개체를 사용하여 기본 항목 수로 값 0을 설정합니다.

5. 파일 메뉴에서 빌드 > 솔루션 빌드를 선택합니다.

ItemCount 속성을 TextBlock 컨트롤에 바인딩하려면

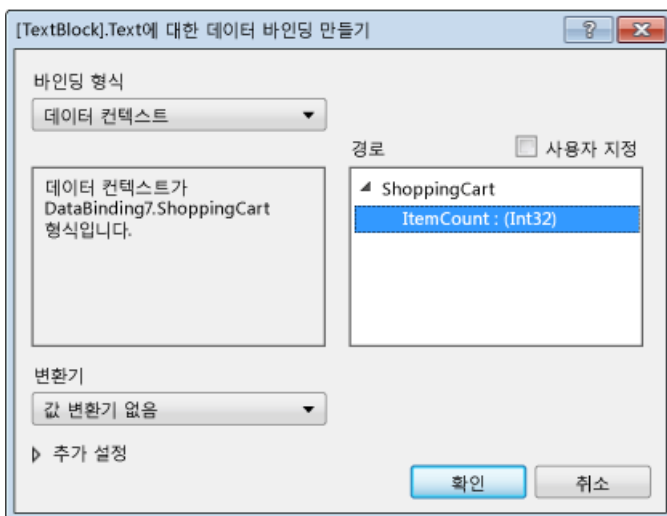
1. 솔루션 탐색기에서 **mainwindow.xaml** 의 바로 가기 메뉴를 열고 **뷰 디자이너**를 선택 합니다.
2. 도구 상자에서 **그리드** 컨트롤을 선택하여 양식에 추가합니다.
3. **Grid**를 선택하고, [속성] 창에서 **DataContext** 속성 옆에 있는 **새로 만들기** 단추를 선택합니다.
4. 개체 선택 대화 상자에서, 모든 어셈블리 표시 확인란을 선택 취소하고, **BindingTest** 네임스페이스 아래에서 **ShoppingCart**를 선택한 다음, **확인** 단추를 선택합니다.

다음 그림에서는 **ShoppingCart**가 선택된 개체 선택 대화 상자를 보여 줍니다.



5. 도구 상자에서, **TextBlock** 컨트롤을 선택하여 양식에 추가합니다.
6. **TextBlock** 컨트롤을 선택하고, [속성] 창에서 **Text** 속성의 오른쪽에 속성 마커를 선택한 다음, **데이터 바인딩 만들기**를 선택합니다. 속성 표시는 작은 상자 모양입니다.
7. [데이터 바인딩 만들기] 대화 상자의 **경로** 상자에서 **ItemCount : (int32)** 속성을 선택한 다음 **확인** 단추를 선택합니다.

다음 그림에서는 **ItemCount** 속성이 선택된 **데이터 바인딩 만들기** 대화 상자를 보여 줍니다.



8. **F5** 키를 눌러 앱을 실행 합니다.

TextBlock 컨트롤은 0의 기본값을 텍스트로 표시해야 합니다.

참고 항목

- XAML 디자이너를 사용하여 UI 만들기
- 값 변환기 추가 대화 상자

XAML 디자이너에서 프로젝트 코드 디버그 또는 사용하지 않도록 설정

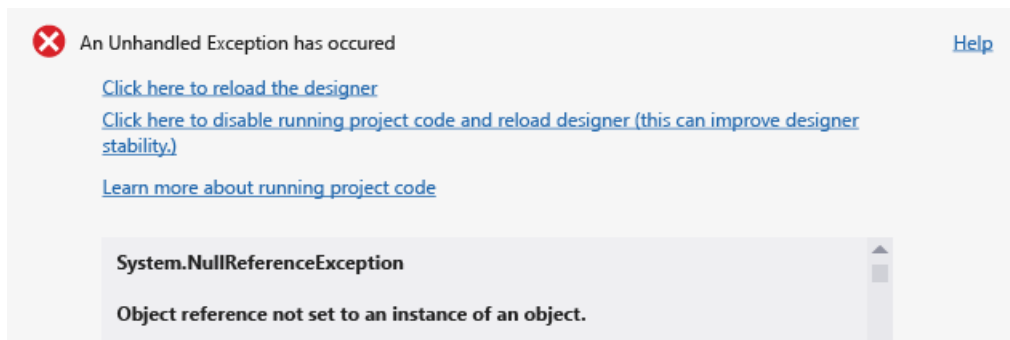
2020-05-08 • 10 minutes to read • [Edit Online](#)

많은 경우 XAML 디자이너의 처리되지 않은 예외는 디자이너에서 애플리케이션이 실행 중일 때 다른 방식으로 작동하거나 다른 값을 반환하는 속성 또는 메서드에 액세스하려는 프로젝트 코드로 인해 발생할 수 있습니다. 이러한 예외는 Visual Studio의 다른 인스턴스에서 프로젝트 코드를 디버그하여 해결하거나 디자이너에서 프로젝트 코드를 사용하지 않도록 설정하여 임시로 방지할 수 있습니다.

프로젝트 코드는 다음을 포함합니다.

- 사용자 지정 컨트롤 및 사용자 지정 컨트롤
- 클래스 라이브러리
- 값 변환기
- 프로젝트 코드에서 생성된 디자인 타임 데이터에 대한 바인딩

프로젝트 코드를 사용할 수 없을 경우 Visual Studio는 자리 표시자를 표시합니다. 예를 들어 Visual Studio에서는 데이터를 더 이상 사용할 수 없는 바인딩에 대한 속성 이름 또는 더 이상 실행하지 않는 컨트롤에 대한 자리 표시자를 보여 줍니다.



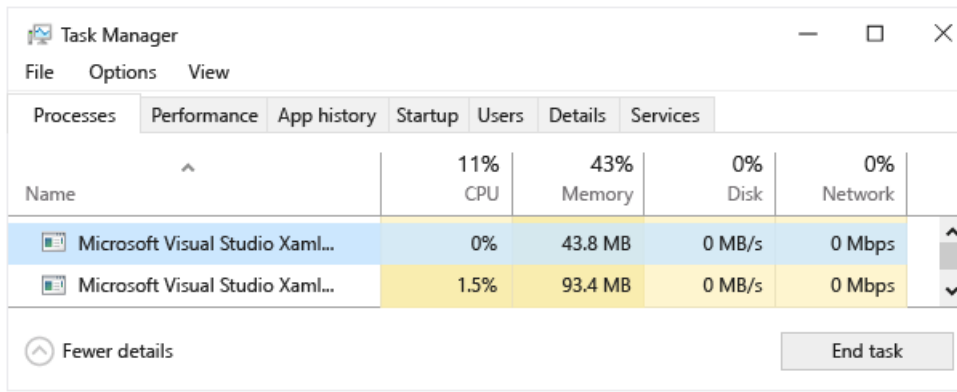
프로젝트 코드가 예외를 발생시키는지 확인하려면

1. 처리되지 않은 예외 대화 상자에서 디자이너를 다시 로드하려면 [여기를 클릭 링크](#)를 선택합니다.
2. 메뉴 모음에서 디버그 > 디버깅 시작 을 선택 하여 응용 프로그램을 빌드하고 실행 합니다.

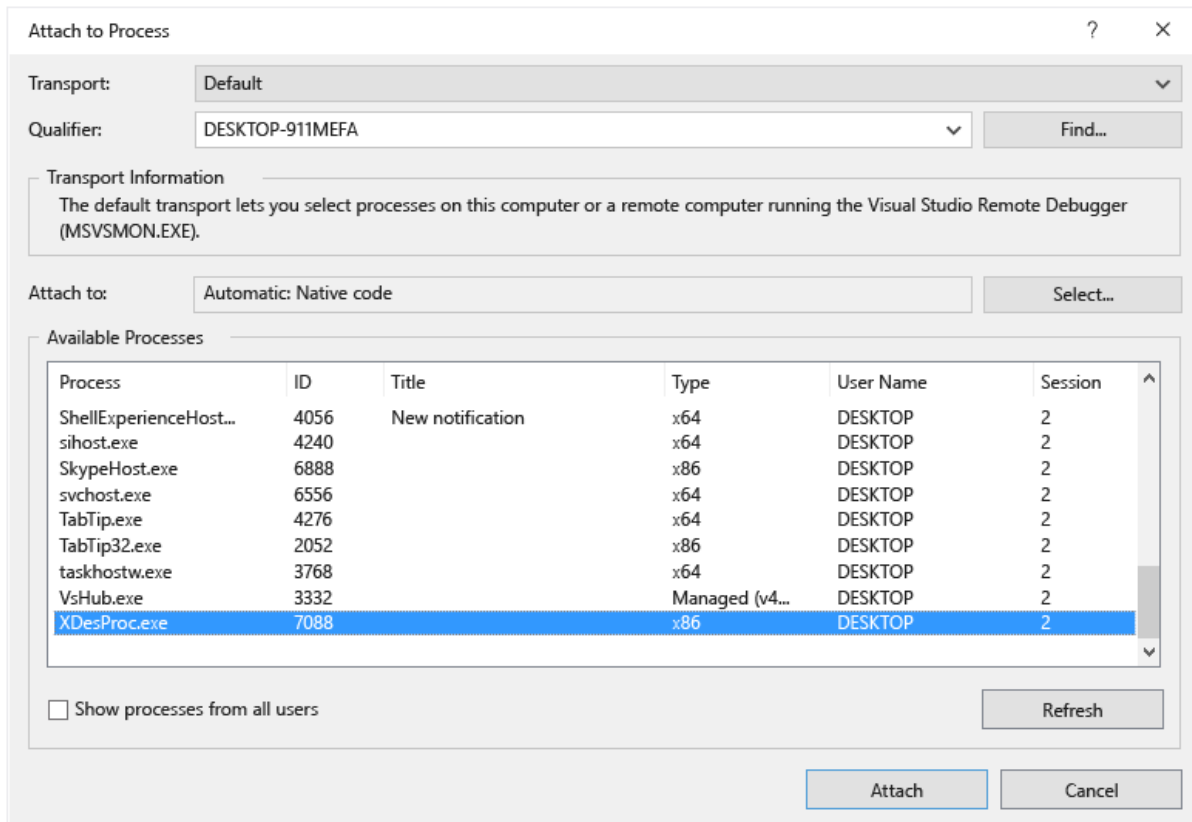
애플리케이션이 성공적으로 빌드되고 실행되면 디자이너에서 실행 중인 프로젝트 코드로 인해 디자인 타임 예외가 발생할 수 있습니다.

디자이너에서 실행되는 프로젝트 코드를 디버그하려면

1. 처리되지 않은 예외 대화 상자에서 프로젝트 코드 실행을 사용하지 않도록 설정하고 디자이너를 다시 로드하려면 [여기를 클릭 링크](#)를 선택합니다.
2. Windows 작업 관리자에서 작업 끝내기 단추를 선택하여 현재 실행 중인 Visual Studio XAML 디자이너의 모든 인스턴스를 닫습니다.



3. Visual Studio에서 디버그하려는 코드 또는 컨트롤이 포함된 XAML 페이지를 엽니다.
4. Visual Studio의 새 인스턴스를 연 다음 프로젝트의 두 번째 인스턴스를 엽니다.
5. 프로젝트 코드에서 중단점을 설정합니다.
6. Visual Studio의 새 인스턴스 메뉴 모음에서 디버그 > 프로세스에 연결을 선택 합니다.
7. 프로세스에 연결 대화 상자의 사용 가능한 프로세스 목록에서 XDesProc.exe를 선택한 다음 연결 단추를 선택합니다.



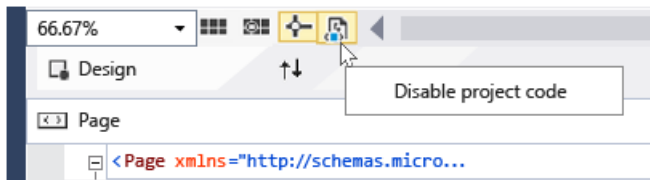
이는 Visual Studio의 첫 번째 인스턴스의 XAML 디자이너에 대한 프로세스입니다.

8. Visual Studio의 첫 번째 인스턴스 메뉴 모음에서 디버그 > 디버깅 시작을 선택 합니다.

이제 디자이너에서 실행 중인 코드를 한 단계씩 실행할 수 있습니다.

디자이너에서 프로젝트 코드를 사용하지 않도록 설정하려면

- 처리되지 않은 예외 대화 상자에서 프로젝트 코드 실행을 사용하지 않도록 설정하고 디자이너를 다시 로드하려면 여기를 클릭 링크를 선택합니다.
- 또는 XAML 디자이너의 도구 모음에서 프로젝트 코드 사용 안 함 단추를 선택 합니다.



단추를 다시 토글하여 프로젝트 코드를 다시 사용하도록 설정할 수 있습니다.

NOTE

ARM 또는 X64 프로세서를 대상으로 하는 프로젝트의 경우 Visual Studio가 디자이너에서 프로젝트 코드를 실행할 수 없으므로 디자이너에서 프로젝트 코드를 사용하지 않도록 설정 단추가 사용하지 않도록 설정됩니다.

- 두 옵션 모두 디자이너가 다시 로드한 다음, 연결된 프로젝트에 대한 모든 코드를 사용하지 않도록 설정합니다.

NOTE

프로젝트 코드를 사용하지 않도록 설정하면 디자인 타임 데이터가 손실될 수 있습니다. 그러므로 디자이너에서 실행되는 코드를 디버그하는 것이 좋습니다.

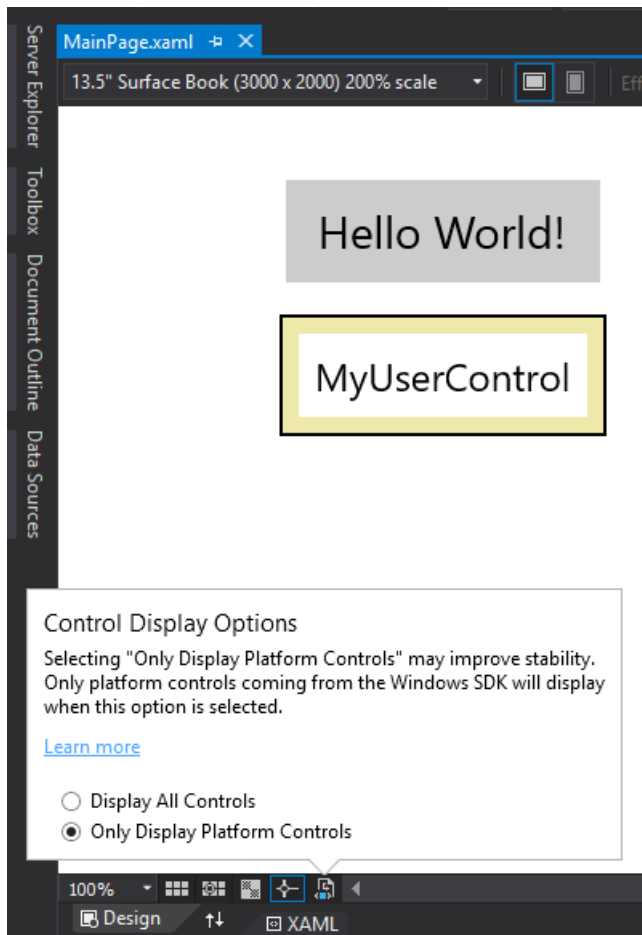
컨트롤 표시 옵션

NOTE

컨트롤 표시 옵션은 Windows 10 Fall Creators Update(빌드 16299) 이상을 대상으로 하는 유니버설 Windows 플랫폼 애플리케이션에만 사용할 수 있습니다. 컨트롤 표시 옵션 기능은 Visual Studio 2017 버전 15.9 이상에서 사용할 수 있습니다.

XAML 디자이너에서 Windows SDK의 플랫폼 컨트롤만 표시하도록 컨트롤 표시 옵션을 변경할 수 있습니다. 이로 인해 XAML 디자이너의 안정성이 향상될 수 있습니다.

컨트롤 표시 옵션을 변경하려면 디자이너 창의 왼쪽 아래에서 아이콘을 클릭하고 컨트롤 표시 옵션 아래의 옵션을 선택합니다.



플랫폼 컨트롤만 표시를 선택하면 SDK, 고객 사용자 정의 컨트롤 등에서 제공되는 모든 사용자 지정 컨트롤이 완벽하게 렌더링되지 않습니다. 대신, 대체 컨트롤로 바뀌어 컨트롤의 크기와 위치를 보여 줍니다.

참고 항목

- [Visual Studio 및 Blend for Visual Studio에서 XAML 디자인](#)

Blend for Visual Studio 개요

2020-05-08 • 8 minutes to read • [Edit Online](#)

Blend for Visual Studio를 사용하면 XAML 기반 Windows 및 웹 애플리케이션을 디자인할 수 있습니다. Visual Studio와 같은 기본 XAML 디자인 환경을 제공하고, 애니메이션 및 동작과 같은 고급 작업에 대한 비주얼 디자이너를 추가합니다. Blend 및 Visual Studio 비교를 보려면 [Visual Studio 및 Blend for Visual Studio에서 XAML 디자인을 참조하세요](#).

Blend for Visual Studio는 Visual Studio의 구성 요소입니다. Blend를 설치하려면 **Visual Studio 설치 관리자**에서 유니버설 **Windows 플랫폼 개발** 또는 **.NET 데스크톱 개발** 워크로드를 선택합니다. 이러한 워크로드에는 둘 다 Blend for Visual Studio 구성 요소가 포함됩니다.

Summary

- > Visual Studio core editor
- ✓ Universal Windows Platform development
 - Included
 - ✓ Blend for Visual Studio
 - ✓ .NET Native and .NET Standard
 - ✓ NuGet package manager
 - ✓ Universal Windows Platform tools
 - ✓ Windows 10 SDK (10.0.16299.0) for UWP: C#, VB, JS

Summary

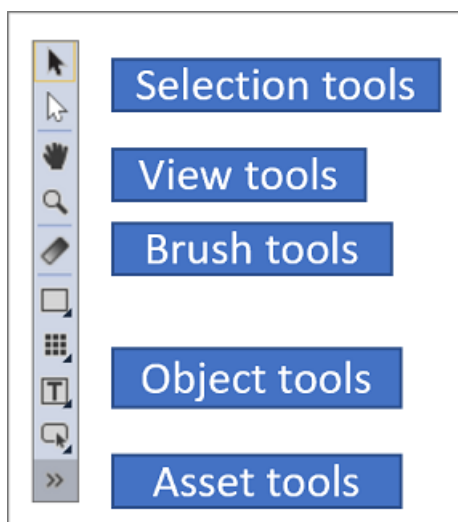
- > Visual Studio core editor
- ✓ .NET desktop development
 - Included
 - ✓ .NET desktop development tools
 - ✓ .NET Framework 4.6.1 development tools
 - ✓ C# and Visual Basic
 - Optional
 - ✓ .NET Framework 4 – 4.6 development tools
 - ✓ Blend for Visual Studio
 - ✓ Entity Framework 6 tools
 - ✓ .NET profiling tools
 - ✓ Just-In-Time debugger
 - ☐ F# desktop language support

Blend for Visual Studio를 처음 접하는 경우 시간을 두고 작업 영역의 고유 기능을 익히세요. 이 항목에 이러한 작업 영역이 간단히 설명되어 있습니다.

도구 패널

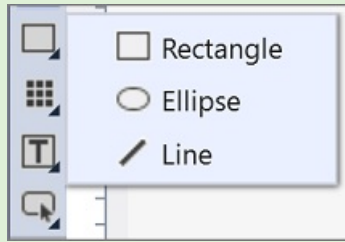
Blend for Visual Studio에서 도구 패널을 사용하여 애플리케이션의 개체를 만들고 수정할 수 있습니다. `.xaml` 파일이 열려 있을 때 도구 패널이 XAML 디자이너의 왼쪽에 표시됩니다.

도구를 선택한 다음 마우스로 아트보드에 그려 개체를 만들 수 있습니다.



TIP

도구 패널의 일부 도구에는 사각형이 아닌 변형이 있습니다. 예를 들어 타원이나 선을 선택할 수 있습니다. 이러한 변형에 액세스하려면 도구를 마우스 오른쪽 단추로 클릭하거나 누른 상태를 유지합니다.



선택 도구

개체 및 패스를 선택합니다. **직접 선택** 도구는 중첩된 개체 및 패스 세그먼트를 선택하는 데 사용됩니다.

뷰 도구

이동, 확대/축소 등 아트보드의 뷰를 조정합니다.

브러시 도구

브러시 변형, 그라데이션 적용 등 개체의 시각적 특성으로 작업합니다.

개체 도구

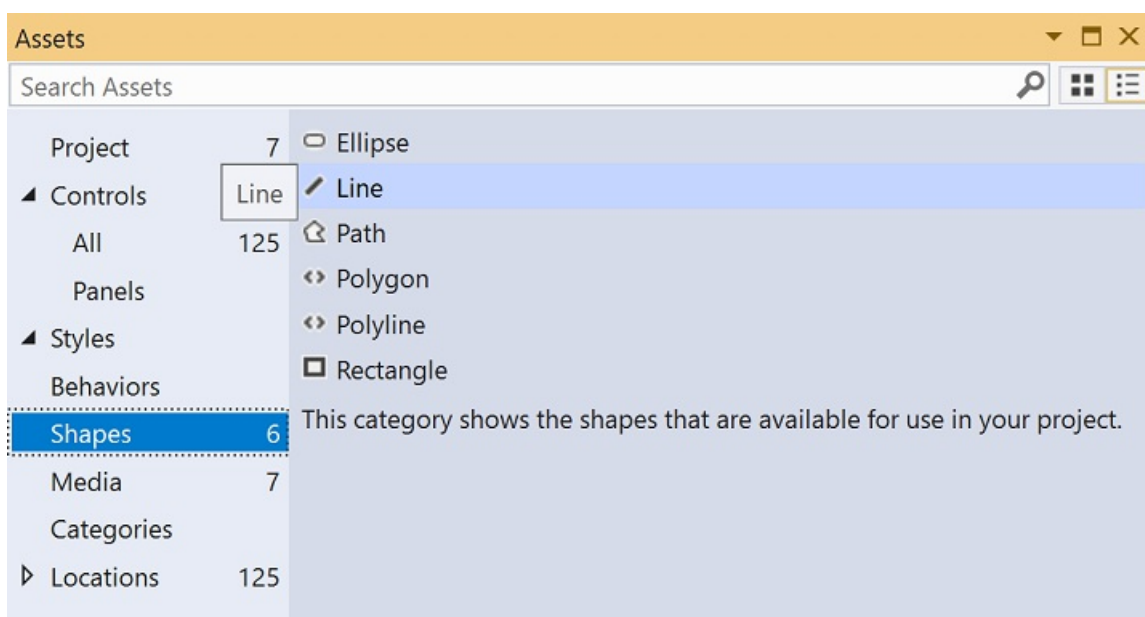
아트보드에서 패스, 도형, 레이아웃 패널, 텍스트, 컨트롤 등 가장 일반적으로 사용되는 개체를 그리는 데 사용됩니다.

자산 도구

자산 창에 액세스하고 라이브러리에서 가장 최근에 사용된 자산을 표시하는 데 사용됩니다.

자산 창

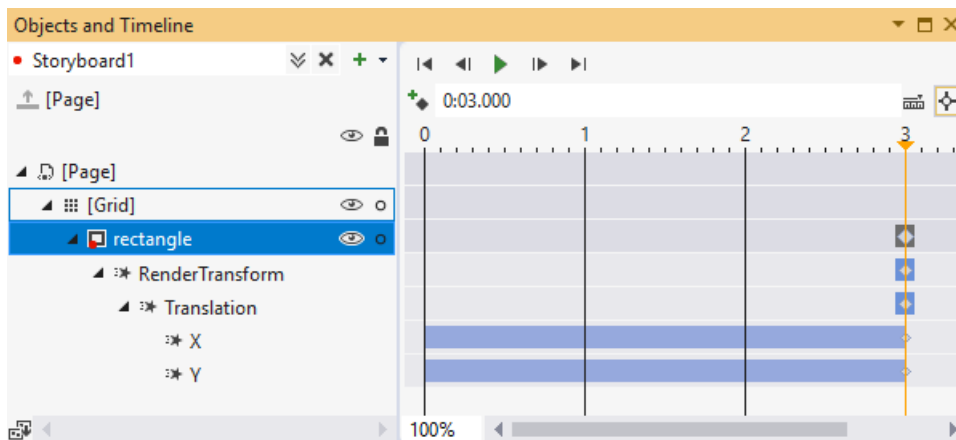
자산 창에는 사용 가능한 모든 컨트롤이 포함되어 있으며 Visual Studio의 도구 상자와 유사합니다. 컨트롤 외에도, 스타일, 미디어, 동작, 효과 등 무엇이든 **자산** 창의 아트보드에 추가할 수 있습니다. **자산** 창을 열려면 **보기 > 자산 창**을 선택하거나 **Ctrl+Alt+X**를 누릅니다.

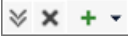


- 자산 검색 상자에 텍스트를 입력하여 자산 목록을 필터링합니다.
- 오른쪽 위에 있는 단추를 사용하여 자산의 Grid 모드와 List 모드 뷰 보기 간에 전환합니다.

개체 및 타임라인 창

이 창에서는 아트보드에서 개체를 구성하고 원하는 경우 개체에 애니메이션 효과를 적용할 수 있습니다. 개체 및 타임라인 창을 열려면 **보기 > 문서 개요**를 선택합니다. Visual Studio의 [문서 개요 창](#)에 제공된 기능 외에도 Blend for Visual Studio의 개체 및 타임라인 창의 오른쪽에는 타임라인 컴퍼지션 영역이 있습니다. 애니메이션을 만들고 편집할 때 타임라인을 사용합니다.



스토리보드 관련 단추 사용  스토리보드를 만들거나 삭제하거나 닫거나 선택합니다. 오른쪽의 타임라인 컴퍼지션 영역을 사용하여 타임라인을 확인하고 키 프레임을 이동합니다.

사용 가능한 기능에 대한 자세한 내용을 보려면 창의 각 단추를 마우스로 가리킵니다.

참고 항목

- [개체에 애니메이션 적용](#)
- [도형 및 패스 그리기](#)
- [Visual Studio 및 Blend for Visual Studio에서 XAML 디자인](#)

도형 및 패스 그리기

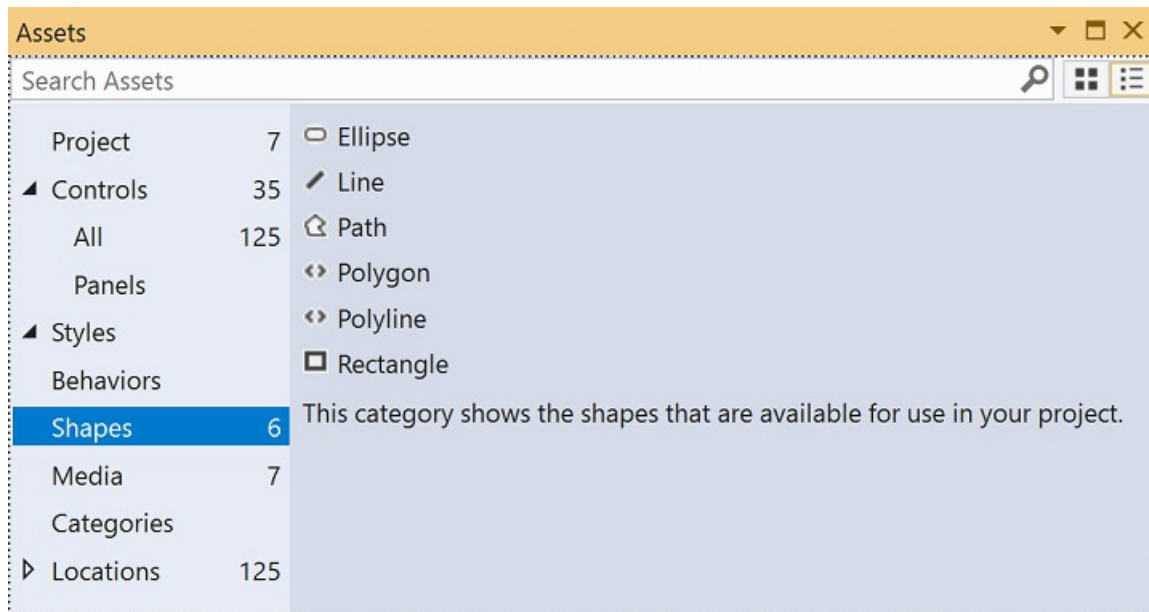
2020-05-08 • 8 minutes to read • [Edit Online](#)

XAML 디자이너에서 **도형**은 정확히 필요한 것입니다. (예: 사각형, 원, 또는 타원). **패스**는 도형의 보다 유연한 버전으로 도형의 모양을 변경하거나 도형을 결합하는 등 작업을 수행하여 새 도형을 만들 수 있습니다.

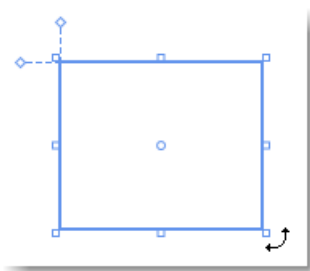
도형 및 패스는 벡터 그래픽을 사용하여 고해상도 디스플레이에 맞게 조정됩니다.

도형 그리기

자산 창에서 도형을 찾습니다.



아트보드에 원하는 모든 도형을 끕니다. 그런 다음 도형에서 핸들을 사용하여 비율 크기를 조정하고 셰이프를 회전, 이동하거나 기울일 수 있습니다.



패스 그리기

패스는 일련의 연결된 선 및 곡선입니다. 패스를 사용하여 **자산** 창에서 사용할 수 없는, 흥미로운 도형을 만들 수 있습니다.

선, 펜 또는 연필을 사용하여 패스를 그릴 수 있습니다. 이러한 도구는 **도구** 창에 있습니다.

직선 그리기

펜 도구나 줄 도구를 사용합니다.

펜 도구 사용

아트보드에서 한 번 클릭하여 시작 점을 정의한 후 다시 클릭하여 줄의 끝을 정의합니다.

줄 도구 사용

아트보드에서 줄을 시작할 위치에서 마우스를 끌어 줄을 끝낼 지점에서 마우스를 놓습니다.

곡선 그리기

펜 도구를 사용합니다.

아트보드에서 한 번 클릭하여 줄의 시작 점을 정의한 후 마우스 포인터를 클릭한 상태에서 끌어 원하는 곡선을 만듭니다.

패스를 닫으려면 줄에서 처음 지점을 클릭합니다.

곡선의 모양 변경

직접 선택 도구를 사용합니다.

도형을 클릭하고 도형에서 아무 점을 마우스로 끌어 곡선 모양을 변경합니다.

자유형 패스 그리기

연필 도구를 사용합니다.

아트보드에서 실제 연필을 사용하는 것처럼 자유형 패스를 그립니다.

패스의 일부 제거

직접 선택 도구를 사용합니다.

삭제할 세그먼트가 있는 패스를 선택한 후 **삭제** 단추를 클릭합니다.

패스에서 점 제거

선택 도구를 사용하여 경로를 선택 합니다. 그런 다음 **펜** 도구를 사용하여 제거하려는 점을 클릭합니다.

패스에 점 추가

선택 도구를 사용하여 경로를 선택 합니다. **펜** 도구를 사용하여 패스에서 점을 추가할 위치를 클릭합니다.

도형을 패스로 변환

패스를 수정하는 방법과 같은 방법으로 도형을 수정하려면 도형을 패스로 변환합니다. 셰이프를 선택한 다음 **서식 > 경로 > 패스로 변환**을 선택 합니다.

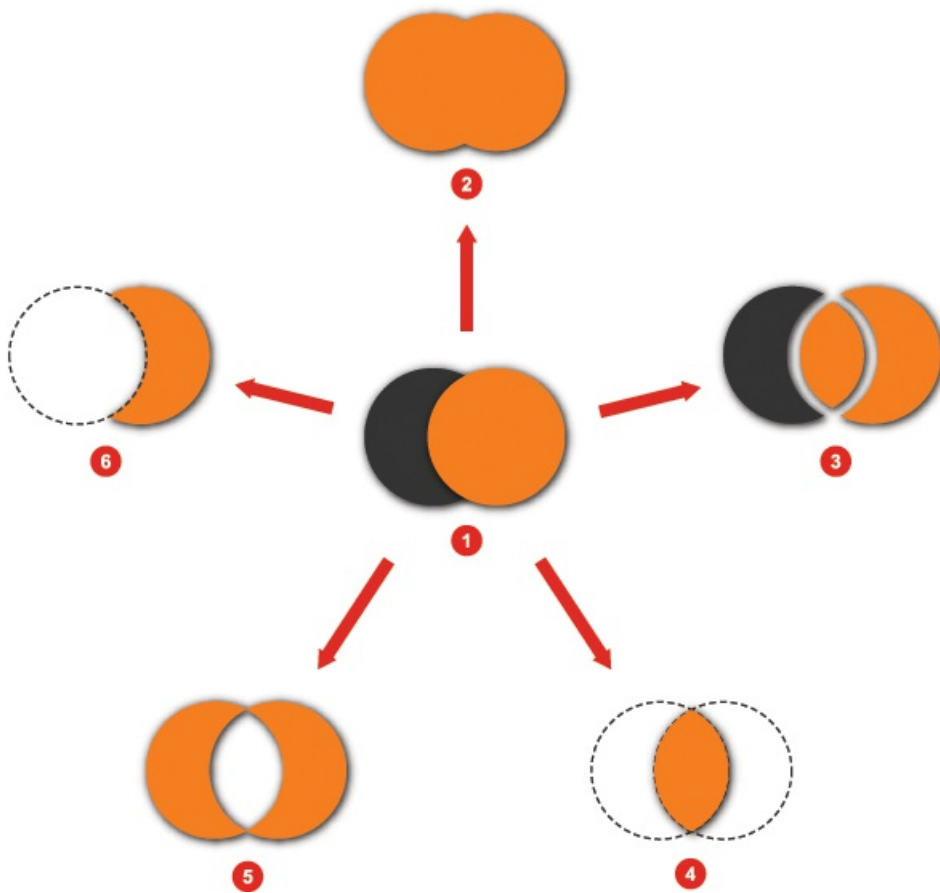
짧은 비디오 보기:  [경로 사용: 도형을 경로로 변환.](#)

NOTE

패스로 변환은 최소 `TargetPlatformVersion` 10.0.16299.0 이상인 UWP 앱에는 현재 사용할 수 없습니다.

패스 결합

패스 및 도형을 하나의 패스로 결합할 수 있습니다.



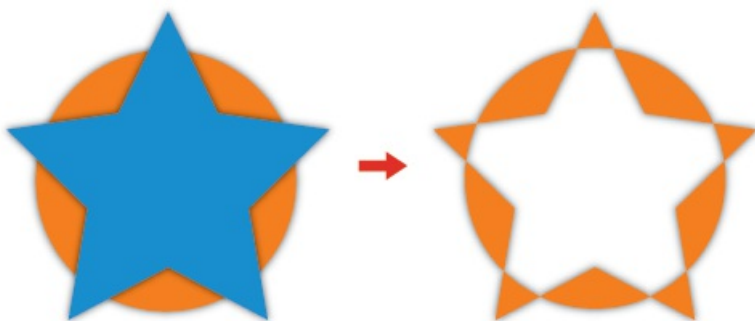
1	결합하기 전의 두 도형	4	Intersect
2	통합	5	겹침 제외
3	나누기	6	빼기

짧은 비디오 보기: [▶ 경로 사용: 경로 결합.](#)

복합형 패스 만들기

복합형 패스를 만들 때 패스의 교차되는 부분은 결과에서 제외되며, 결과 패스는 맨 아래 패스의 시각적 속성을 사용합니다.

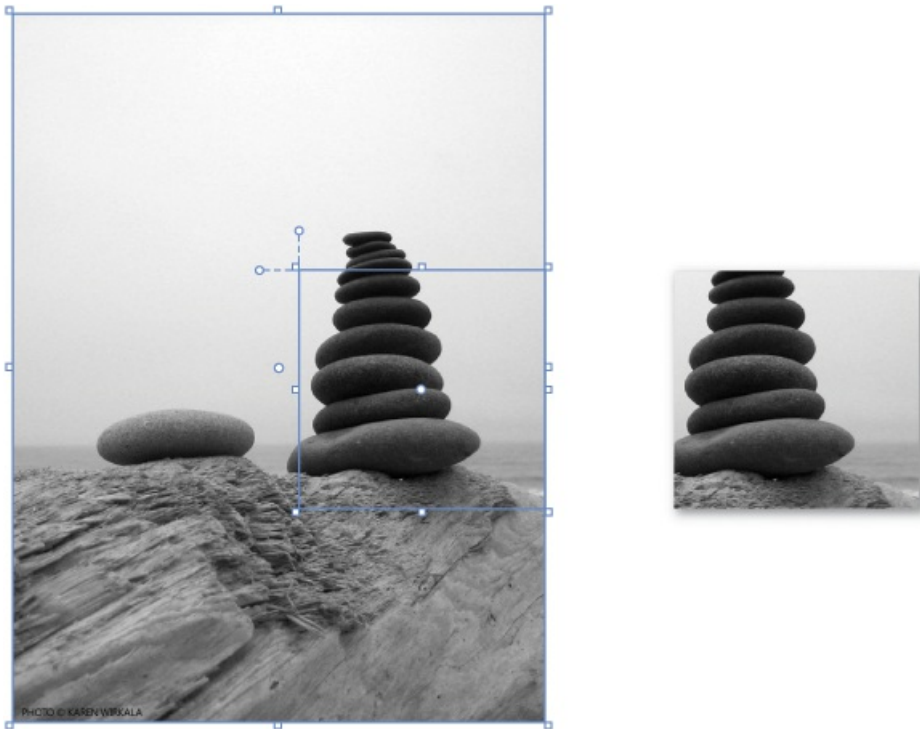
복합형 패스를 만든 후 언제든지 분리할 수 있습니다.



짧은 비디오 보기: [▶ 패스 사용: 복합형 패스 만들기.](#)

클리핑 패스 만들기

클리핑 패스는 다른 개체에 적용되는 패스나 도형이며, 개체에서 클리핑 패스를 벗어나는, 마스크된 개체 부분을 숨깁니다.



짧은 비디오 보기:  패스 사용: 클리핑 패스 만들기.

Blend for Visual Studio에서 개체 스타일 수정

2020-05-08 • 12 minutes to read • [Edit Online](#)

개체를 사용자 지정하는 가장 쉬운 방법은 속성 창에서 속성을 설정하는 방법입니다.

설정이나 설정 그룹을 다시 사용하려면 다시 사용할 수 있는 리소스를 만듭니다. 이러한 리소스로는 *스타일*, *템플릿* 또는 사용자 지정 색 등을 들 수 있습니다. 또한 해당 상태에 따라 컨트롤이 다르게 표시되도록 할 수 있습니다. 예를 들어 사용자가 단추를 클릭할 때 단추가 녹색으로 바뀝니다.

브러시: 개체의 모양 수정

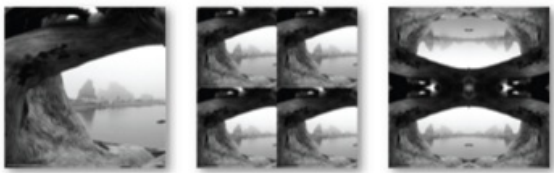
모양을 변경하려는 경우 브러시를 개체에 적용합니다.

개체에 반복되는 이미지 또는 패턴 그리기

*타일 브러시*를 사용하여 개체에 반복되는 이미지나 패턴을 그릴 수 있습니다.

타일 브러시를 만들려면 *이미지 브러시*, *드로잉 브러시* 또는 *비주얼 브러시* 리소스를 만들어 시작합니다.

이미지를 사용하여 이미지 브러시를 만듭니다. 다음 그림은 이미지 브러시, 바둑판식 이미지 브러시 및 대칭 이동 이미지 브러시를 보여 줍니다.



패스나 도형 등과 같은 벡터 드로잉을 사용하여 드로잉 브러시를 만듭니다. 다음 그림은 드로잉 브러시, 바둑판식 드로잉 브러시 및 대칭 이동 드로잉 브러시를 보여 줍니다.



컨트롤에서 단추 등과 같은 비주얼 브러시를 만듭니다. 다음 그림은 비주얼 브러시, 바둑판식 비주얼 브러시를 보여 줍니다.



스타일 및 템플릿: 컨트롤 간 일관된 모양과 느낌 만들기

컨트롤의 모양 및 동작을 한 번 디자인하고 해당 디자인을 다른 컨트롤에 적용하여 컨트롤의 모양과 동작을 개별적으로 유지 관리하지 않아도 됩니다.

스타일을 사용해야 하나요?: 기본 속성(예: 단추 색)을 설정하려면 *스타일*을 사용합니다. 스타일을 컨트롤에 적용한 후에도 컨트롤을 수정할 수 있습니다.

템플릿을 사용해야 하나요?: 컨트롤의 구조를 변경하려는 경우 *템플릿*을 사용합니다. 그래픽이나 로고를 단추로

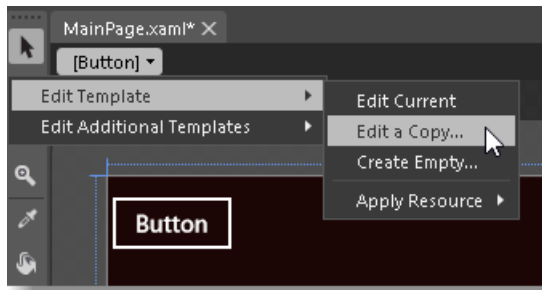
변환한다고 가정해 보면 템플릿을 컨트롤에 적용한 후에는 컨트롤을 수정할 수 없습니다.

템플릿 또는 스타일 만들기

두 가지 방법으로 템플릿을 만들 수 있습니다. 아트보드의 모든 개체를 컨트롤로 변환하거나 기존 컨트롤의 템플릿을 기준으로 설정할 수 있습니다.

모든 개체를 컨트롤 템플릿으로 변환하려면 개체를 선택한 후 도구 메뉴에서 **컨트롤로 만들기**를 선택합니다.

기존 컨트롤의 템플릿을 기준으로 설정하려면 아트보드에서 개체를 선택합니다. 그런 다음 아트보드 상단에서 이동 경로 탐색 단추를 선택하고 **템플릿 편집**을 선택한 다음 **복사본 편집** 또는 **빈 항목 만들기**를 선택합니다.

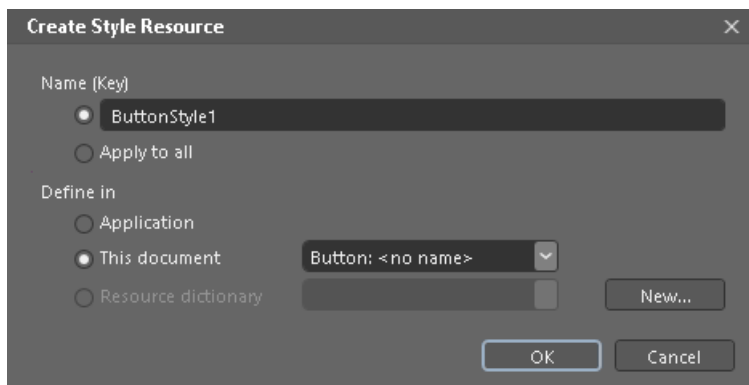


스타일을 만들려면 개체를 선택하고 **개체** 메뉴에서 **스타일 편집**을 선택한 후 **복사본 편집**이나 **빈 항목 만들기**를 선택합니다.


- 컨트롤의 기본 스타일이나 템플릿으로 시작하려면 **복사본 편집**을 선택합니다.
- 처음부터 다시 시작하려면 **빈 항목 만들기**를 선택합니다.

현재 항목 편집 옵션은 이미 만든 스타일이나 템플릿을 편집하는 경우에만 표시됩니다. 기본 시스템 템플릿을 사용하는 컨트롤에 대해서는 표시되지 않습니다.

스타일 리소스 만들기 대화 상자에서 나중에 사용할 수 있도록 스타일이나 템플릿의 이름을 지정하거나 해당 유형의 모든 컨트롤에 스타일이나 템플릿을 적용할 수 있습니다.

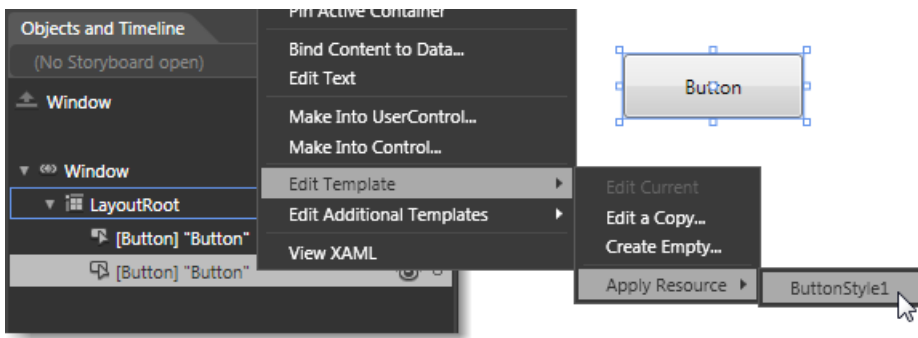


NOTE

일부 유형의 컨트롤에 대해서는 스타일이나 템플릿을 만들 수는 없습니다. 컨트롤에서 스타일이나 템플릿을 지원하지 않는 경우 아트보드 위에 이동 경로 탐색 단추가 표시되지 않습니다. 기본 문서의 편집 범위로 돌아오려면 **범위 되돌리기** 을 클릭합니다.

컨트롤에 스타일이나 템플릿 적용

개체 및 타임라인 창에서 개체를 마우스 오른쪽 단추로 클릭하고 **템플릿 편집**을 선택한 다음, **리소스 적용**을 선택합니다.

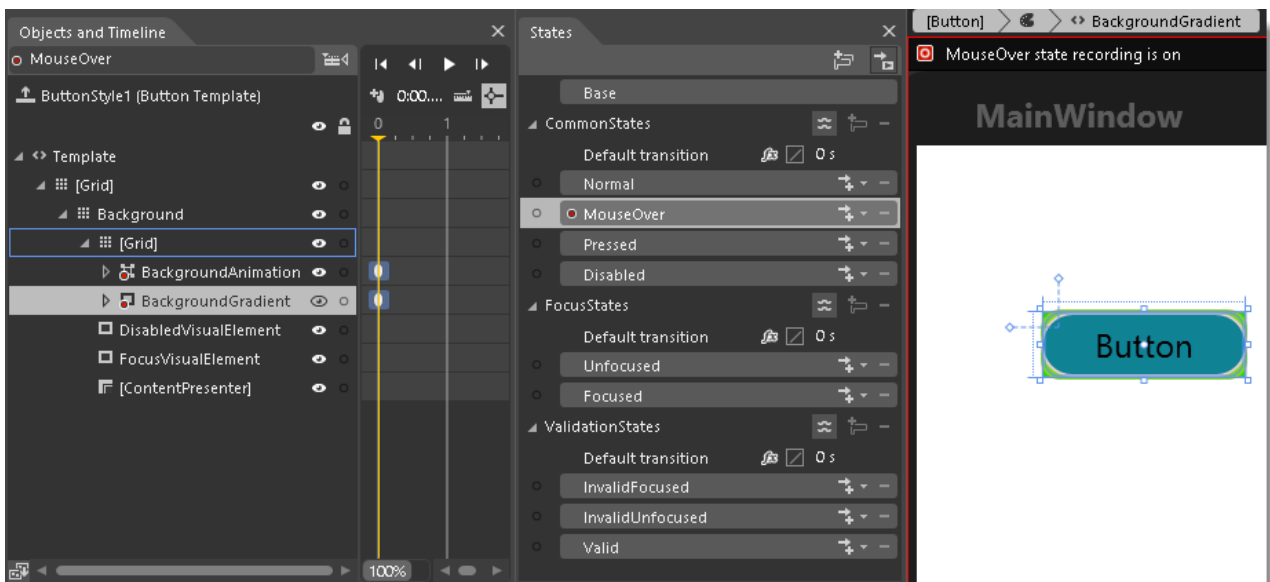


컨트롤의 기본 스타일이나 템플릿 복원

컨트롤을 선택 하고 ** 속성 **** 창에서 스타일 또는 템플릿 속성을 찾습니다. 고급 옵션을 선택한 다음, 바로 가기 메뉴에서 다시 설정을 클릭합니다.

시각적 상태

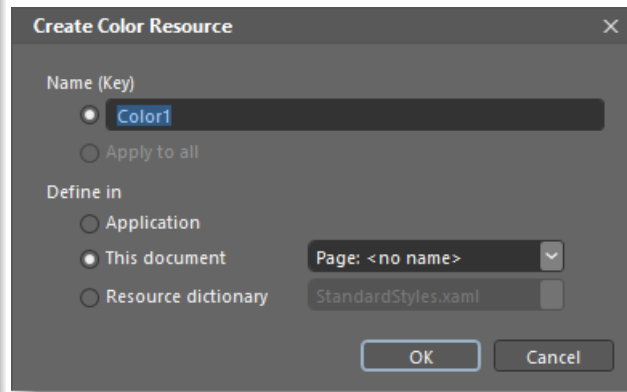
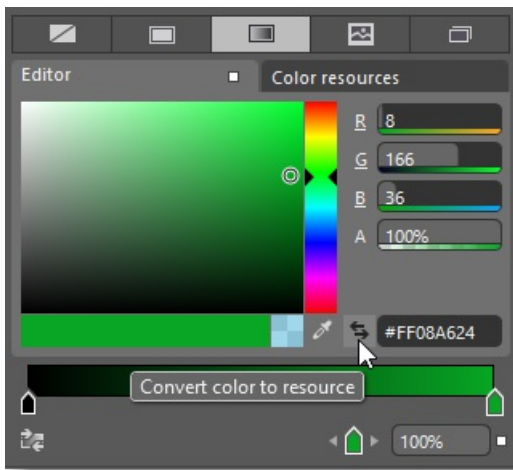
시각적 상태를 사용하여 해당 상태에 따라 컨트롤의 모양을 변경할 수 있습니다. 컨트롤에는 사용자 상호 작용에 따라 여러 시각적 모양이 포함될 수 있습니다. 예를 들어 사용자가 클릭하거나 애니메이션 실행 시 단추가 녹색으로 바뀌도록 설정할 수 있습니다. 전환을 사용하여 시각적 상태 간의 시간을 줄이거나 늘릴 수 있습니다.



짧은 비디오 보기: [Manage the state of your WPF controls](#)(WPF 컨트롤 상태 관리).

리소스: 색, 스타일 및 템플릿을 만들고 나중에 다시 사용

프로젝트의 모든 항목을 리소스로 변환할 수 있습니다. 리소스는 애플리케이션의 다른 위치에 다시 사용할 수 있는 단순한 개체입니다. 예를 들어 색을 한 번 만들어 리소스로 만든 후 해당 색을 여러 개체에 사용할 수 있습니다. 이러한 모든 개체의 색을 변경하려면 색 리소스를 변경하기만 하면 됩니다.



참고 항목

- [Blend for Visual Studio를 사용하여 UI 만들기](#)

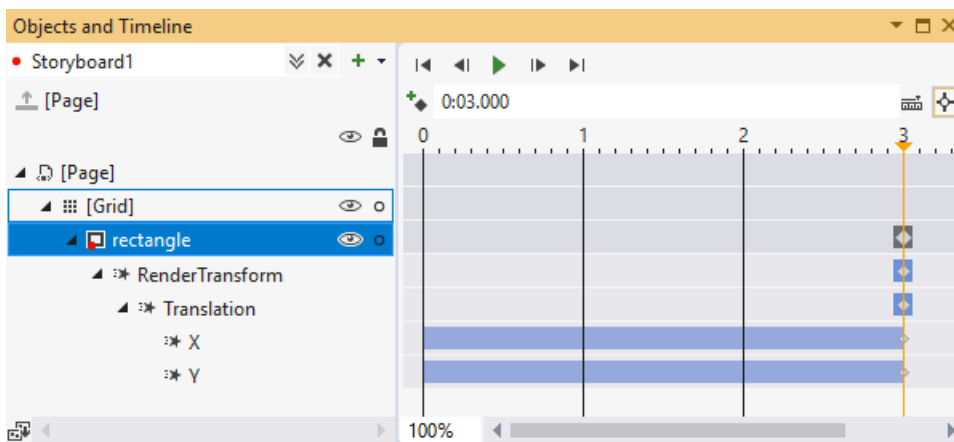
XAML 디자이너에서 개체에 애니메이션 효과 주기

2020-05-08 • 4 minutes to read • [Edit Online](#)

예를 들어 Blend for Visual Studio를 사용하면 개체를 이동하거나 페이드 인 및 페이드 아웃하는 짧은 애니메이션을 쉽게 만들 수 있습니다.

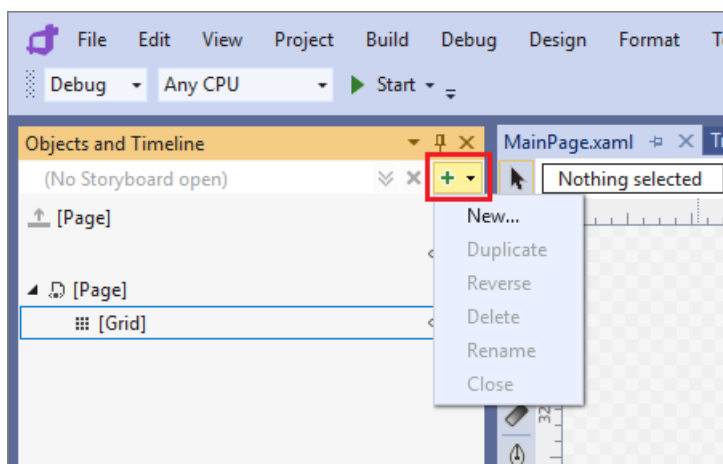
애니메이션을 만들려면 *스토리보드*가 필요합니다. 스토리보드에는 하나 이상의 *타임 라인*이 포함됩니다. 타임라인에서 *키 프레임*을 설정하여 속성 변경 내용을 표시합니다. 그런 다음 애니메이션을 실행하면 Blend for Visual Studio가 지정된 기간 동안 속성 변경 내용을 보간합니다. 따라서 부드러운 전환이 가능해집니다. 비시각적 속성을 비롯하여 개체에 속하는 모든 속성에 애니메이션 효과를 적용할 수 있습니다.

다음 이미지는 **Storyboard1**이라는 스토리보드를 보여 줍니다. 타임라인에는 사각형의 X 및 Y 위치를 표시하는 키 프레임이 포함됩니다. 이 애니메이션을 실행하면 사각형이 특정 위치에서 다른 위치로 부드럽게 이동합니다.

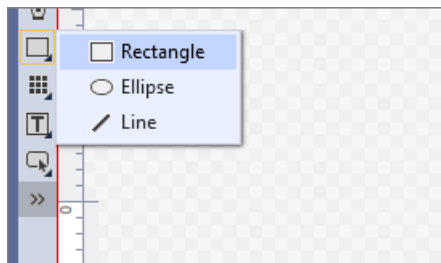


애니메이션 만들기

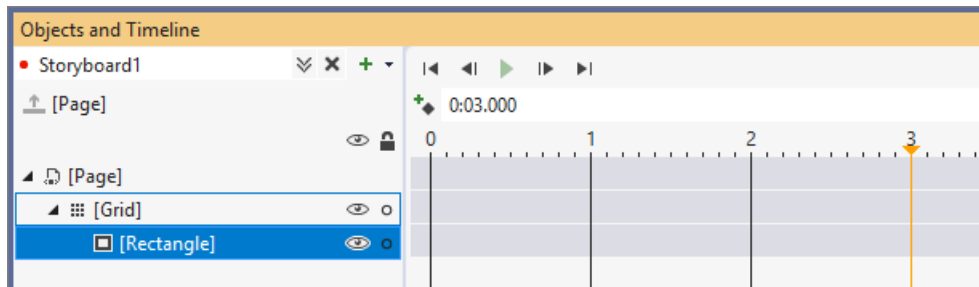
1. 스토리보드를 만들려면 개체 및 타임라인 창에서 스토리보드 옵션 단추를 선택한 후 새로 만들기를 선택합니다.



2. 스토리보드 만들기 리소스 대화 상자에서 스토리보드의 이름을 입력합니다.
3. 디자인 뷰의 자산 패널에서 페이지 왼쪽 아래에 사각형을 추가합니다.



4. 개체 및 타임라인 창에서 노란색 시간 포인터를 3 초로 이동합니다.



5. 페이지의 디자인 뷰에서 사각형을 페이지의 오른쪽으로 끌어 놓습니다.
6. 재생을 누르면 사각형이 페이지의 왼쪽에서 오른쪽으로 이동합니다.

시점에 따라 사각형의 다른 변경 내용으로 재생합니다. 예를 들어 채우기 색을 변경하거나 속성 창에서 방향을 대칭 이동할 수 있습니다.

참고 항목

- [Blend for Visual Studio를 사용하여 UI 만들기](#)

Blend for Visual Studio에서 데이터 표시

2020-05-08 • 3 minutes to read • [Edit Online](#)

페이지의 레이아웃을 사용자 지정할 때 디자이너에서 예제 데이터를 볼 수 있습니다. 예제 데이터는 처음부터 새로 또는 기존 클래스를 사용하여 생성할 수 있습니다. 예제 데이터를 실행시 앱에 표시되는 *라이브 데이터*에 연결할 수도 있습니다.

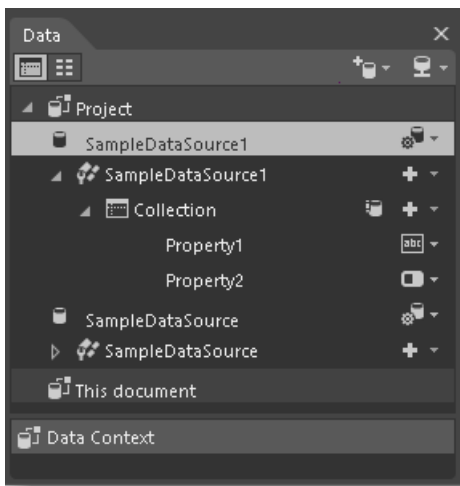
NOTE

Blend의 데이터 패널은 .NET Framework를 대상으로 하는 프로젝트에만 지원 됩니다. .NET Core를 대상으로 하는 UWP 프로젝트 또는 프로젝트에 대해서는 지원 되지 않습니다.

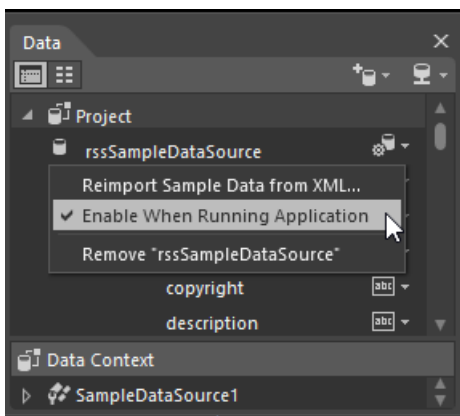
샘플 데이터 생성

예제 데이터를 생성하려면 XAML 문서를 엽니다. 데이터 패널에서 **예제 데이터 만들기** 단추를 클릭한 다음, 새 예제 데이터를 선택합니다.

데이터 패널에서 데이터 구조를 정의한 다음 모든 페이지의 UI 요소에 바인딩합니다.




앱 실행 시 예제 데이터가 페이지에 표시되도록 하려면 데이터 원본 옵션을 선택한 다음, 애플리케이션 실행 시 사용을 선택합니다.



짧은 비디오 보기: [▶ 처음부터 예제 데이터 만들기.](#)

클래스에서 예제 데이터 생성

데이터 구조를 설명하는 클래스를 이미 만들었으면 이 클래스에서 예제 데이터를 생성할 수 있습니다.

클래스에서 예제 데이터를 생성하려면 XAML 문서를 열고 데이터 패널에서 **예제 데이터 만들기**  단추를 클릭한 다음, 클래스에서 **예제 데이터 만들기**를 클릭합니다.

짧은 비디오 보기:  [일부 데이터 바인딩을 Blend와 혼합](#).

참고 항목

- [Blend for Visual Studio를 사용하여 UI 만들기](#)

XAML 오류 및 경고

2020-05-08 • 2 minutes to read • [Edit Online](#)

XAML을 작성하면 Visual Studio에서는 입력한 코드를 분석합니다. 오류가 검색되면 오류 표시선이 코드 줄에 나타납니다. 하지만 오류 표시선 위로 마우스를 가져가면 오류 또는 경고에 대한 자세한 정보를 제공합니다. 일부 오류 및 경고의 경우 빠른 작업 전구가 표시 되고 **Ctrl+**를 사용 합니다. 바로 가기 키는 문제를 해결하는 옵션을 표시합니다.

오류 형식

내부적으로 여러 도구는 병렬로 XML을 분석합니다. XML 오류는 오류를 검색하는 도구에 따라 다음 세 가지 형식 중 하나로 분류됩니다.

검색된 오류 기준	오류 코드 형식
XAML 언어 서비스(XAML 편집기)	XLStxxx
XAML 디자이너	XDGxxxx
XAML 편집하며 계속하기	XECxxxx

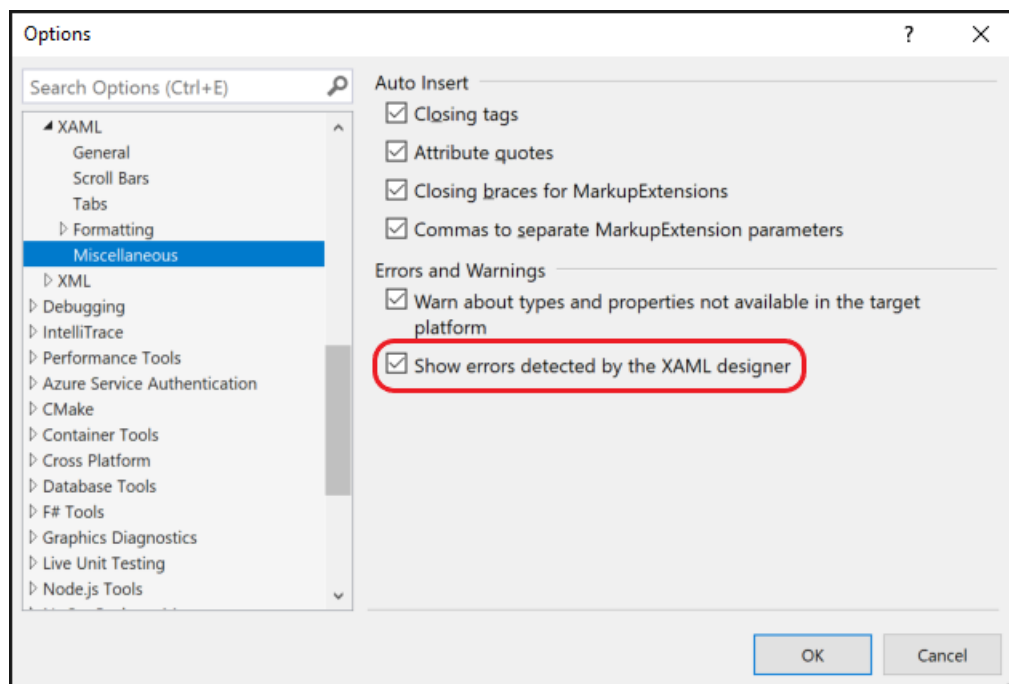
NOTE

일부 오류 또는 경고에는 해당 코드가 포함됩니다. 이러한 오류는 일반적으로 XML 디자이너 오류입니다.

XAML 디자이너 오류 표시 안 함

도구 > 옵션을 선택하여 옵션 대화 상자를 연 다음, 텍스트 편집기 > XML > 기타를 선택합니다.

XAML 디자이너에 의해 검색된 오류 표시 확인란의 선택을 취소합니다.



XAML 디자이너에 대한 바로 가기 키

2020-05-08 • 4 minutes to read • [Edit Online](#)

XAML 디자이너의 바로 가기 키를 사용하면 마우스 단추를 여러 번 클릭해야 하는 동작을 바로 가기 키 하나로 줄일 수 있으므로 작업 속도를 향상시킬 수 있습니다.

요소 바로 가기

이 표에서는 아트 보드의 요소 작업에 사용할 수 있는 바로 가기 키를 보여줍니다.

수행할 작업	수행할 작업
요소 만들기	Ctrl+N 누르기
요소 복제	Alt 키를 누른 상태로 화살표 키를 누릅니다.
컨트롤의 텍스트 편집	F2 키를 누릅니다. 종료하려면 Esc 키를 누릅니다.
단일 요소 선택	Tab 키를 눌러 문서에 나타나는 순서 대로 요소를 선택합니다. 화살표 키를 사용하여 요소를 선택할 수도 있습니다.
여러 요소 선택	Shift 키를 누른 상태에서 각 요소를 선택합니다.
인접하지 않은 여러 요소 선택	Ctrl 키를 누른 상태에서 첫 번째와 마지막 요소를 선택합니다.
선택한 요소 이동	화살표 키를 누릅니다. Shift 키를 누른 상태에서 키 누름당 이동할 거리를 늘릴 수 있습니다.
15도씩 요소 회전	Shift 키를 누른 상태에서 요소를 회전합니다.
모든 요소 선택	Ctrl+A 를 누릅니다.
모든 요소 선택 지우기	Ctrl+Shift+A 를 누릅니다.
요소 핸들 표시 또는 숨기기	F9 키를 누릅니다.
요소의 속성 선택	요소를 선택 하고 속성 창에 포커스를 두고 tab 키를 누릅니다. (Ctrl+Tab 을 사용 하여 포커스를 속성 창로 변경 합니다.) 화살표 키를 사용 하여 드롭다운 목록에서 속성 값을 선택할 수 있습니다.

문서 개요 창 바로 가기

다음 표에서는 문서 개요 창에서 요소 작업을 할 때 사용 가능한 바로 가기 키를 나열합니다.

수행할 작업	수행할 작업
포커스가 문서 개요 창에 있을 때 아트보드 개체 숨기기	Ctrl+H

수행할 작업	수행할 작업
포커스가 문서 개요 창에 있을 때 아트보드 개체 표시	Shift+키 Ctrl+H
포커스가 문서 개요 창에 있을 때 아트보드 개체 잠금	Ctrl+L
포커스가 문서 개요 창에 있을 때 아트보드 개체 잠금 해제	Shift+CtrlCtrl+L

참고 항목

- [XAML 디자이너를 사용하여 UI 만들기](#)

Blend for Visual Studio의 바로 가기 키

2020-05-08 • 9 minutes to read • [Edit Online](#)

프로젝트 바로 가기

원하는 작업	단계
새 프로젝트 만들기	Ctrl+ShiftShift+N
프로젝트 또는 솔루션(사이트 아님) 열기	Ctrl+ShiftShift+O
솔루션 닫기	Ctrl+ShiftShift+C
솔루션 또는 사이트의 복사본 저장	Ctrl+ShiftShift+P
프로젝트에 기존 항목 추가	Ctrl+I
DLL에 참조 추가(WPF)	Alt+ShiftShift+R
프로젝트 빌드	Ctrl+ShiftShift+B
프로젝트 또는 사이트 테스트	F5

문서 바로 가기

원하는 작업	단계
열린 문서 간 전환	Ctrl+Tab
활성 문서 저장	Ctrl+S
모든 문서 저장	Ctrl+ShiftShift+S
활성 문서 닫기	Ctrl+W
열린 문서 모두 닫기	Ctrl+ShiftShift+W
마지막 작업 실행 취소	Ctrl+ Z
실행 취소한 마지막 작업 다시 실행	Ctrl+Y 또는 Ctrl+Shift+Z
디자인 타임 주석 만들기	Ctrl+ShiftShift+T
잘라내기	Ctrl+X
복사	Ctrl+C
붙여넣기	Ctrl+V

원하는 작업	단계
DELETE	삭제
텍스트 찾기(XAML 뷰 또는 JavaScript 편집기에만 해당)	Ctrl+F
텍스트의 다음 항목 찾기(XAML 뷰 또는 JavaScript 편집기에만 해당)	F3 또는 Ctrl+H

개체 바로 가기

원하는 작업	단계
새 항목 만들기	Ctrl+N
개체 복제	Alt 키 를 누른 채로 개체 끌기
개체 부모 재지정	Alt 키를 누른 상태에서 개체를 레이아웃 패널 위로 끌기
컨트롤의 텍스트 편집	F2 (끝내려면 Esc)
컨트롤 편집(WPF)	Ctrl+E
선택한 개체의 너비를 같게 만들기	Ctrl+ShiftShift+1
선택한 개체의 높이를 같게 만들기	Ctrl+ShiftShift+2
선택한 개체의 크기를 같게 만들기	Ctrl+ShiftShift+9
선택한 개체를 가로로 대칭 이동	Ctrl+ShiftShift+3
선택한 개체를 세로로 대칭 이동	Ctrl+ShiftShift+4
여러 개체 선택	Ctrl 키 를 누른 채
여러 인접한 개체 선택	Shift 키 를 누른 채
선택 영역을 화면 크기에 맞추기	Ctrl+9
활성 컨테이너 고정	Ctrl+ShiftShift+D
선택한 개체 조금 이동	화살표 키
너비 자동 조정	Ctrl+ShiftShift+5
높이 자동 조정	Ctrl+ShiftShift+6
개체를 레이아웃 패널로 그룹화	Ctrl+G
개체 그룹 해제	Ctrl+ShiftShift+G
선택한 개체를 맨 앞으로 가져오기	Ctrl+ShiftShift+]

원하는 작업	단계
앞으로 가져오기	Ctrl+]
선택한 개체를 뒤로 보내기	Ctrl+ShiftShift+[
뒤로 보내기	Ctrl+[
선택한 개체에서 사용자 컨트롤 만들기(WPF)	F8
개체의 비율 제한	Shift 키를 누른 채로 개체 끌기
개체를 15도씩 회전	Shift 키를 누른 채로 개체 회전
클리핑 패스 만들기	Ctrl+7
클리핑 패스 해제	Ctrl+ShiftShift+7
복합형 패스 만들기	Ctrl+8
복합형 패스 해제	Ctrl+ShiftShift+8
선택 항목 잠금	Ctrl+L
모든 개체 잠금 해제	Ctrl+ShiftShift+L
선택 항목 표시	Ctrl+T
선택영역 숨기기	Ctrl+3
모든 개체 선택	Ctrl+A
모든 개체 선택 지우기	Ctrl+ShiftShift+A

보기 바로 가기

원하는 작업	단계
디자인, 코드 및 분할 보기 간 전환	F11
아트보드 확대	Ctrl+등호 (=)
아트보드 축소	Ctrl+빼기 기호 (-)
아트보드 확대/축소	마우스 휠 돌리기
아트보드를 왼쪽/오른쪽으로 이동	Shift 키를 누른 채로 마우스 휠 돌리기
아트보드를 위/아래로 이동	Ctrl 키를 누른 채로 마우스 휠 돌리기
선택 영역을 화면 크기에 맞추기	Ctrl+0

원하는 작업	단계
아트 보드를 실제 크기로 보기	Ctrl+1
핸들 표시/숨기기	F9
개체 경계선 표시 또는 숨기기	Ctrl+ShiftShift+H
디자인, XAML 및 분할 보기 간 전환	F11

작업 영역 바로 가기

원하는 작업	단계
애니메이션 작업 영역과 디자인 작업 영역 간 전환	Ctrl+F11
자산 패널 표시/숨기기	Ctrl+마침표
결과 패널 표시/숨기기	F12
패널 모두 표시/숨기기	F4
사용 중인 작업 영역 레이아웃 다시 설정	Ctrl+ShiftShift+R
작업 영역 이동	스페이스바를 누른 채로 유지
다른 도구가 선택되어 있는 동안 임시로 선택 도구 사용	Ctrl 키 를 누른 채

Blend의 아트보드 보조 키

2020-05-08 • 4 minutes to read • [Edit Online](#)

일부 바로 가기 키는 연결된 메뉴 항목이 없으므로 Blend 사용자 인터페이스를 통해 찾을 수 없습니다. 다음 표에는 개체 크기 조정 등의 작업을 수정하는 바로 가기가 나열되어 있습니다.


수행할 작업	단계
다른 도구가 선택된 상태에서 일시적으로 선택 도구를 선택합니다. 이렇게 하면 선택 도구와 다른 도구 간에 전환하며 도구 패널에서 클릭해야 하는 횟수가 줄어듭니다.	Ctrl 키를 누른 채
선택 도구를 선택한 상태에서 선택한 개체 조금 이동	화살표 키 누름
아트보드 이동	스페이스바 를 누른 채로 아트보드 끌기
아트보드 확대/축소	마우스 휠 돌리기
아트보드 확대	아트 보드에서 아무 곳이나 클릭 하여 Ctrl+스페이스바 를 누르고 있습니다.
아트보드 축소	아트 보드에서 아무 곳이나 클릭 하는 동안 Ctrl+Alt+스페이스바 를 누르고 있습니다.
아트보드를 왼쪽 및 오른쪽으로 이동	Shift 키를 누른 채로 마우스 휠 돌리기
아트보드를 위/아래로 이동	Ctrl 키를 누른 채로 마우스 휠 돌리기
그리거나 변환 중인 개체의 비율 제한	Shift 키를 누른 채
개체를 15도씩 회전	Shift 키를 누른 채
개체 복제	Alt 키를 누른 채로 개체 끌기
개체 부모 재지정	개체를 레이아웃 패널 위로 끌고 Alt 키를 누른 상태에서 마우스 단추 놓기
여러 개체 선택	Ctrl 키를 누른 채로 각 개체 선택
여러 인접한 개체 선택	Shift 키를 누른 채로 첫 번째 및 마지막 개체 선택
움직이는 텍스트를 그려서 선택	Shift 키를 누른 채로 끌기
다른 개체 아래에 있는 개체 선택	Alt 키를 누른 채로 개체의 각 레이어에 대해 한 번 클릭
열린 문서 간 전환	Ctrl+tab 키를 누릅니다.
자산 패널 열기	Ctrl+Period 누르기

참고 항목

- [바로 가기 키](#)
- [펜 도구 보조 키](#)
- [직접 선택 도구 보조 키](#)

Blend for Visual Studio의 펜 도구 보조 키

2020-05-08 • 6 minutes to read • [Edit Online](#)

다음 표에는 **펜** 도구 를 사용하여 패스를 만드는 동안 패스를 수정하는 데 사용할 수 있는 바로 가기가 나와 있습니다. **펜** 도구를 사용하여 기존 패스에서 점을 추가 또는 제거하거나 두 개의 기존 패스를 연결할 수 있습니다.

수행할 작업	단계	포인트
직선 세그먼트를 시작할 점 만들기	클릭하여 새 점을 만듭니다.	 펜 포인트
곡선 세그먼트를 시작할 점 만들기	클릭하여 새 점을 만든 후 마우스 단추를 손에서 떼기 전에 탄젠트 핸들을 끌어서 조정합니다.	 펜 포인트
날카로운 모퉁이를 만들도록 부드러운 제약 조건 없이 마지막 탄젠트 조정	클릭하여 새 점을 만든 다음, 마우스 단추에서 손을 떼기 전에 Alt 키를 누릅니다.	 펜 조정 포인트
탄젠트 끝점이 개별적으로 작동하여 날카로운 모퉁이를 만들도록 마지막 탄젠트 분할	클릭하여 새 점을 만든 다음, 마우스 단추에서 손을 떼기 전에 Alt 키를 누른 채로 끕니다.	 펜 조정 포인트
탄젠트 끝점을 새 점 주변에서 15도씩 늘려가며 이동	클릭하여 새 점을 만든 후 마우스 단추를 놓기 전에 Shift+Alt 를 누른 채로 끕니다.	 펜 조정 포인트
끝점의 탄젠트 길이를 0으로 줄이기	끝점을 클릭합니다.	 펜 조정 포인트
기존 패스에 새 점 추가	새 점을 만들 위치의 패스를 클릭합니다.	 펜 삽입 포인트
패스에서 점 제거	기존 점 위를 마우스로 가리키고 클릭합니다.	 펜 삭제 포인트
날카로운 모퉁이가 있는 패스 닫기	시작점을 클릭합니다.	 펜 닫기 포인트
모퉁이에서 부드러운 곡선이 있는 패스 닫기	시작점을 클릭하고 마우스 단추를 손에서 떼기 전에 탄젠트 핸들을 끌어서 수정합니다.	 펜 닫기 포인트


수행할 작업	단계	포인트
두 개의 패스를 연결할 때 날카로운 모 퉁이 만들기	두 개의 경로를 선택하고 펜 도구를 클 릭하고 패스 중 하나의 끝점을 클릭한 후 다른 패스의 끝점을 클릭합니다.	 펜 연결 포인트
두 개의 패스를 연결할 때 부드러운 모 퉁이 만들기	두 개의 경로를 선택하고 펜 도구를 클 릭하고 패스 중 하나의 끝점을 클릭한 후 다른 패스의 끝점을 끕니다.	 펜 연결 포인트
새 패스 만들기	Ctrl 키를 누른 채로 이전 패스의 외부 영역을 클릭하여 이전 패스에 점을 추가 하는 것을 중지한 다음, 새 패스를 시작 하려는 위치를 클릭하거나 이 위치로 끕 니다.	 펜 시작 포인트

참고 항목

- [아트보드 보조 키](#)
- [직접 선택 도구 보조 키](#)
- [도형 및 패스 그리기](#)

Blend for Visual Studio의 직접 선택 도구 보조 키

2020-05-08 • 4 minutes to read • [Edit Online](#)

다음 표에는 직접 선택 도구 를 사용하여 기존 경로의 모양을 수정하는 데 사용할 수 있는 바로 가기가 나와 있습니다. 기존 패스에서 점을 추가 또는 제거하거나 두 개의 기존 패스를 조인하려면 펜 도구를 사용합니다.

수행할 작업	단계	포인터
패스의 점에 대해 탄젠트 핸들 표시	패스의 점을 클릭합니다.	 점 이동 포인터
패스의 점을 이동합니다.	패스의 점을 끕니다.	 점 이동 포인터
패스의 두 점 간 세그먼트에 대해 탄젠트 핸들 표시	패스의 세그먼트를 클릭합니다.	 세그먼트 이동 포인터
패스의 두 점 간 세그먼트를 이동합니다.	패스의 세그먼트를 끕니다.	 세그먼트 이동 포인터
패스의 점에 대해 탄젠트 각도 변경	패스의 점 또는 세그먼트를 클릭하여 탄젠트 핸들을 표시한 후 탄젠트 끝점 중 하나를 끕니다.	 탄젠트 이동 포인터
점을 날카로운 모서리로 만들거나 탄젠트를 0으로 줄이기	마우스로 점 위를 가리킨 다음, Alt 를 누른 채 점을 클릭합니다.	 점 변환 포인터
날카로운 모퉁이를 부드럽게(또는 이미 부드러운 경우 클릭한 점을 통과할 때 곡선의 각도 변경)	마우스로 점 위를 가리킨 다음, Alt 를 누른 채 점을 끕니다.	 점 변환 포인터
곡선 세그먼트를 직선으로 변경	마우스로 패스의 세그먼트 위를 가리킨 다음, Alt 를 누른 채 세그먼트를 클릭합니다.	 세그먼트 변환 포인터
세그먼트를 가져와 곡선으로 구부려 포인터 위치 관통	마우스로 패스의 세그먼트 위를 가리킨 다음, Alt 를 누른 채 세그먼트를 끕니다.	 세그먼트 변환 포인터
탄젠트의 한쪽 끝을 다른 쪽과 별개로 조정	점 또는 세그먼트를 직접 선택한 다음, Alt 를 누른 채 탄젠트 끝점을 끕니다.	 탄젠트 변환 포인터

참고 항목

- [아트보드 보조 키](#)
- [펜 도구 보조 키](#)
- [도형 및 패스 그리기](#)

디버그하는 동안 XAML 속성 검사

2020-05-08 • 19 minutes to read • [Edit Online](#)

라이브 시각적 트리 및 라이브 속성 탐색기를 사용하여 실행 중인 XAML 코드를 실시간으로 볼 수 있습니다. 이러한 도구는 실행 중인 XAML 애플리케이션의 UI 요소에 대한 트리 뷰를 제공하고 선택한 UI 요소의 런타임 속성을 보여 줍니다.

다음 구성에서 이러한 도구를 사용할 수 있습니다.

앱 유형	운영 체제 및 도구
Windows Presentation Foundation(4.0 이상) 애플리케이션	Windows 7 이상
유니버설 Windows 앱	Windows 10 이상 (windows 10 SDK 포함)

라이브 시각적 트리의 요소 살펴보기

목록 보기 및 단추가 있는 매우 간단한 WPF 애플리케이션을 시작하겠습니다. 단추를 클릭할 때마다 다른 항목이 목록에 추가됩니다. 짝수 번호 항목은 회색으로 표시되고 홀수 번호 항목은 노란색으로 표시됩니다.

프로젝트를 만듭니다.

1. 새 c # wpf 응용 프로그램을 만들고 (파일 > 새로 > 만들기)"c # wpf"를 입력 한 다음 **wpf 앱 (.net Core)** 또는 **wpf 앱 (.NET Framework)** 을 선택 합니다. **TestXAML**로 이름을 지정합니다.
2. 다음과 같이 MainWindow.xaml을 변경합니다.

```
<Window x:Class="TestXAML.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:TestXAML"
    mc:Ignorable="d"
    Title="MainWindow" Height="350" Width="525">
    <Grid>
        <Button x:Name="button" Background="LightBlue" Content="Add Item" HorizontalAlignment="Left"
            Margin="216,206,0,0" VerticalAlignment="Top" Width="75" Click="button_Click"/>
        <ListBox x:Name="listBox" HorizontalAlignment="Left" Height="100" VerticalAlignment="Top"
            Width="100" Margin="205,80,0,0"/>
    </Grid>
</Window>
```

3. MainWindow.xaml.cs 파일에 다음 명령 처리기를 추가합니다.

```

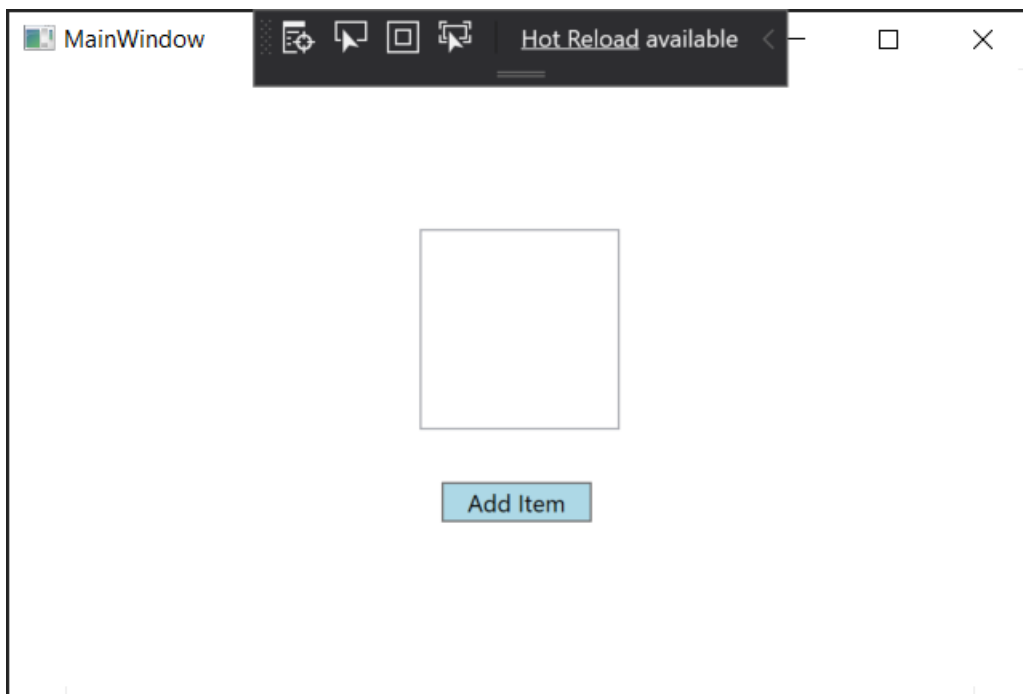
int count;

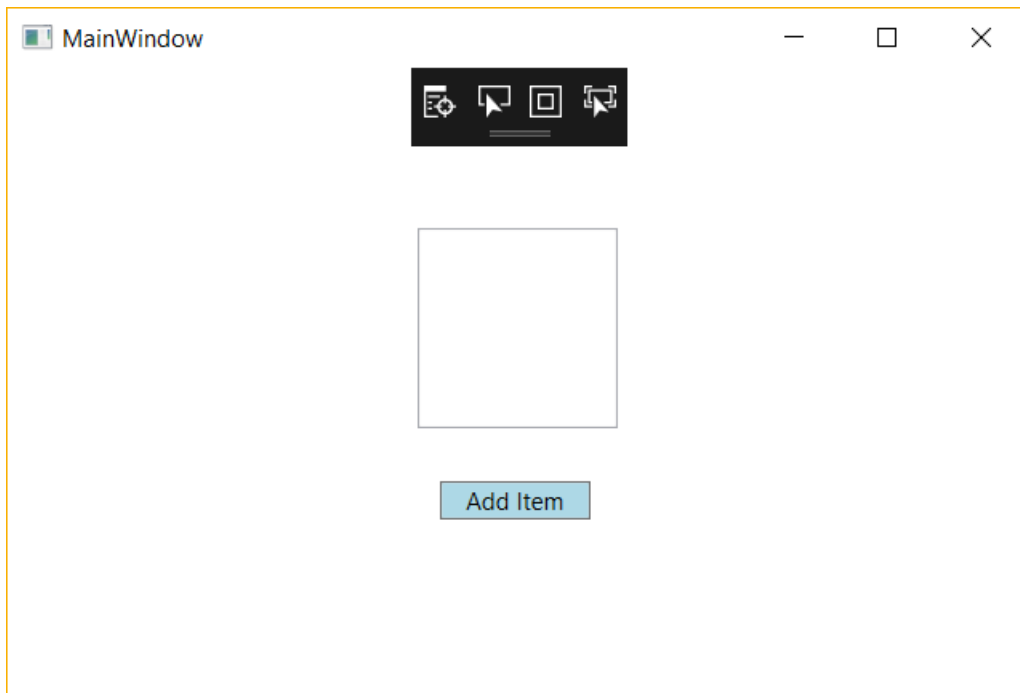
private void button_Click(object sender, RoutedEventArgs e)
{
    ListBoxItem item = new ListBoxItem();
    item.Content = "Item" + ++count;
    if (count % 2 == 0)
    {
        item.Background = Brushes.LightGray;
    }
    else
    {
        item.Background = Brushes.LightYellow;
    }
    listBox.Items.Add(item);
}

```

4. 프로젝트를 빌드하고 디버깅을 시작합니다. (빌드 구성은 릴리스가 아닌 디버그여야 합니다. 빌드 구성에 대한 자세한 내용은 [빌드 구성 이해](#)를 참조 하세요.)

창이 표시 되 면 실행 중인 응용 프로그램 내에 앱 내 도구 모음이 표시 됩니다.





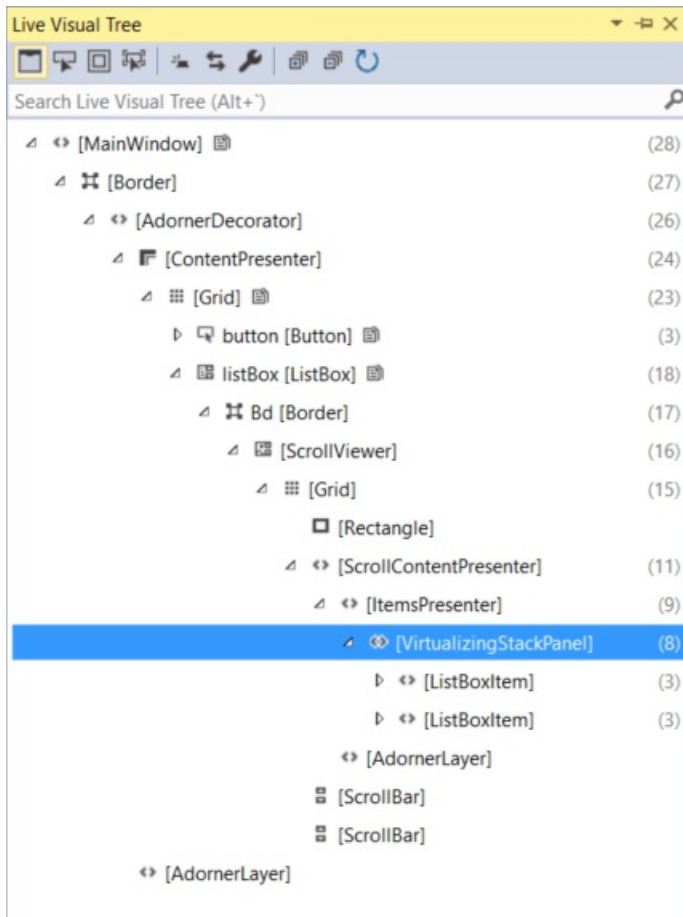
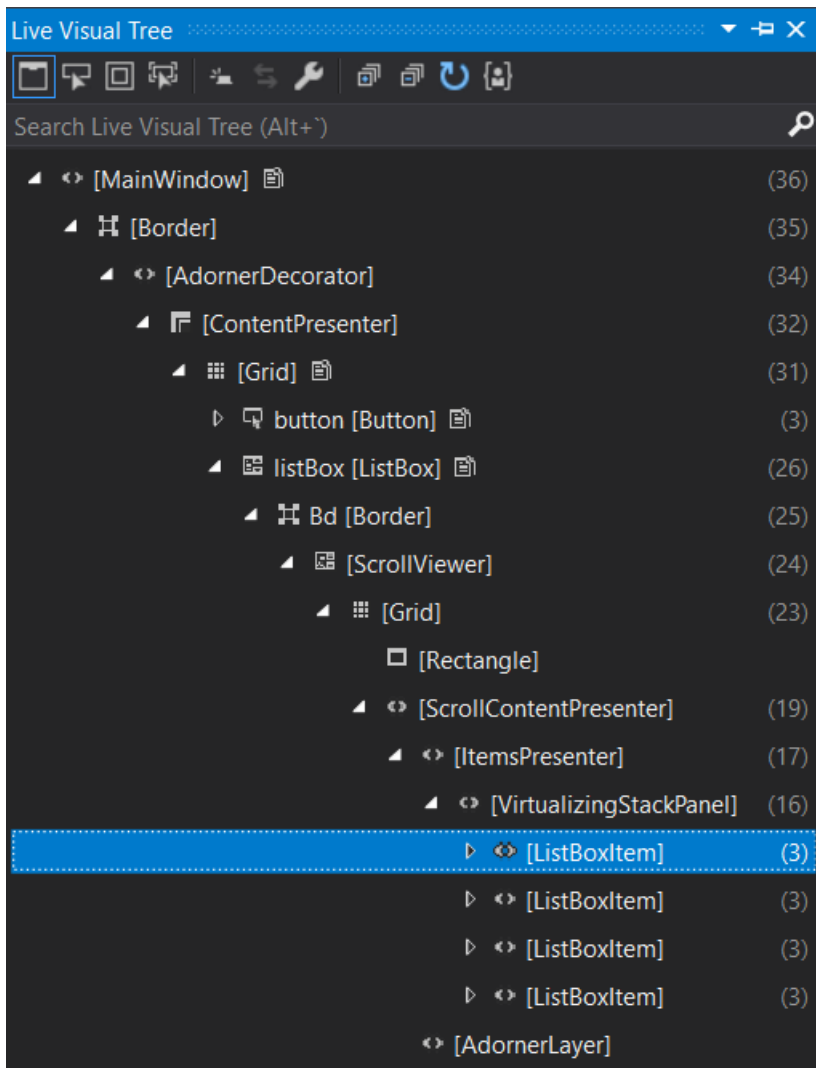
5. 이제 항목 추가 단추를 몇 번 클릭 하여 목록에 새 항목을 추가 합니다.

XAML 속성 검사

1. 그런 다음 앱 내 도구 모음의 왼쪽 단추를 클릭 하거나 디버그 > **Windows** > 라이브 시각적 트리로 이동 하여 **라이브 시각적 트리** 창을 엽니다. 열린 후에는이 창과 **라이브 속성** 창을 나란히 볼 수 있도록 도킹 위치에서 밖으로 끄니다.
2. **라이브 시각적 트리** 창에서 **ContentPresenter** 노드를 확장합니다. 단추와 목록 상자에 대한 노드가 있어야 합니다. 목록 상자와 **ScrollContentPresenter** 및 **ItemsPresenter**를 차례로 확장하여 목록 상자 항목 을 찾습니다.

ContentPresenter 노드가 표시 되지 않으면 도구 모음에서 **내 XAML만 표시** 아이콘을 설정/해제 합니다. Visual Studio 2019 버전 16.4부터 XAML 요소 뷰는 기본적으로 **내 XAML** 기능만 사용 하여 간소화 됩니다. 모든 XAML 요소를 항상 표시 하는 옵션에서 **이 설정을 사용 하지 않도록** 설정할 수도 있습니다.

창이 다음과 같이 표시되어야 합니다.



3. 애플리케이션 창으로 다시 이동하고 더 많은 항목을 추가합니다. 더 많은 목록 상자 항목이 라이브 시각적

트리에 나타나야 합니다.

- 이제 목록 상자 항목 중 하나의 속성을 살펴보겠습니다.

라이브 시각적 트리의 첫 번째 목록 상자 항목을 선택하고 도구 모음에서 속성 표시 아이콘을 클릭합니다. 라이브 속성 탐색기가 표시됩니다. 콘텐츠 필드는 "Item1"이고, 배경색 > 필드는 #FFFFFFE0입니다.

- 라이브 시각적 트리로 돌아가서 두 번째 목록 상자 항목을 선택합니다. 라이브 속성 탐색기는 내용 필드가 "Item2"이고 배경색 > 필드가 #FFD3D3D3 (테마에 따라)임을 표시합니다.

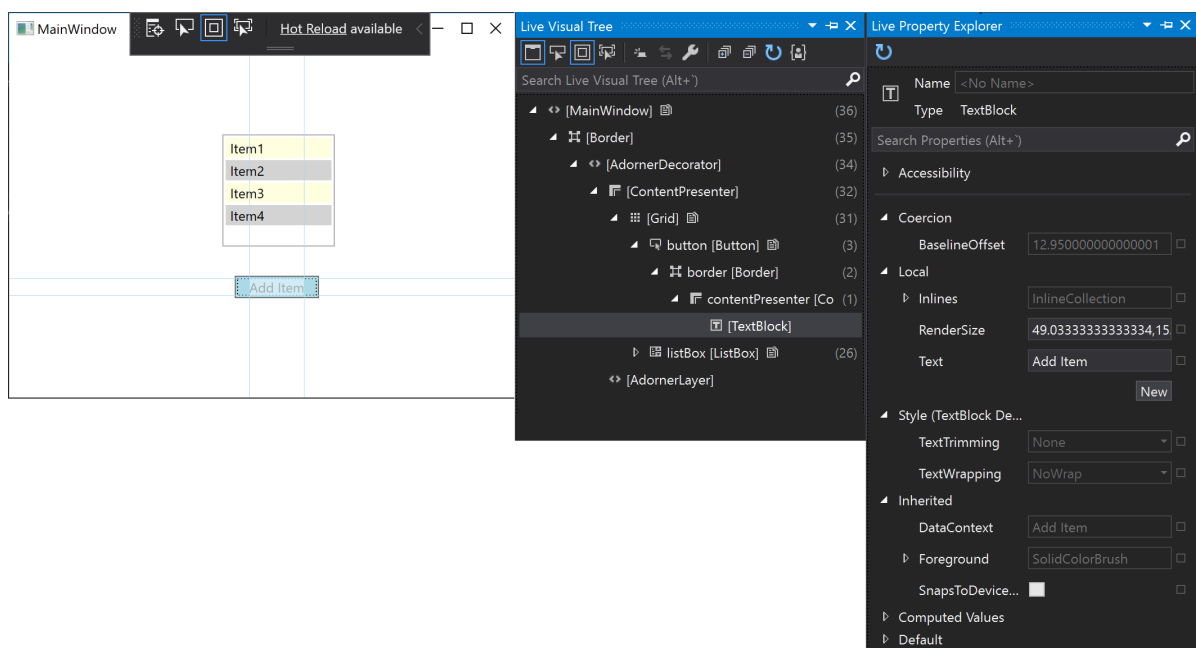
NOTE

라이브 속성 탐색기에서 속성 주위의 노란색 테두리는 속성 값이와 `Color = {BindingExpression}` 같은 바인딩을 통해 설정됨을 의미합니다. 녹색 테두리는와 `Color = {StaticResource MyBrush}` 같은 리소스를 사용하여 값을 설정하는 것을 의미합니다.

XAML의 실제 구조에는 직접적으로 관련이 없는 많은 요소가 있습니다. 코드를 잘 모르면 트리에서 필요한 항목을 찾기가 어려울 수 있습니다. 따라서 라이브 시각적 트리에는 검사하려는 요소를 찾는 데 도움이 되는 애플리케이션 UI를 사용할 수 있는 몇 가지 방법이 있습니다.

실행 중인 응용 프로그램에서 요소를 선택합니다. 라이브 시각적 트리 도구 모음의 가장 왼쪽에 있는 단추를 선택하면 이 모드를 사용할 수 있습니다. 이 모드를 켜면 애플리케이션에서 UI 요소를 선택할 수 있고 라이브 시각적 트리(및 라이브 속성 뷰어)가 이 요소 및 해당 속성에 해당하는 트리에 노드를 표시하도록 자동으로 업데이트됩니다. Visual Studio 2019 버전 16.4부터 **요소 선택의 동작을 구성**할 수 있습니다.

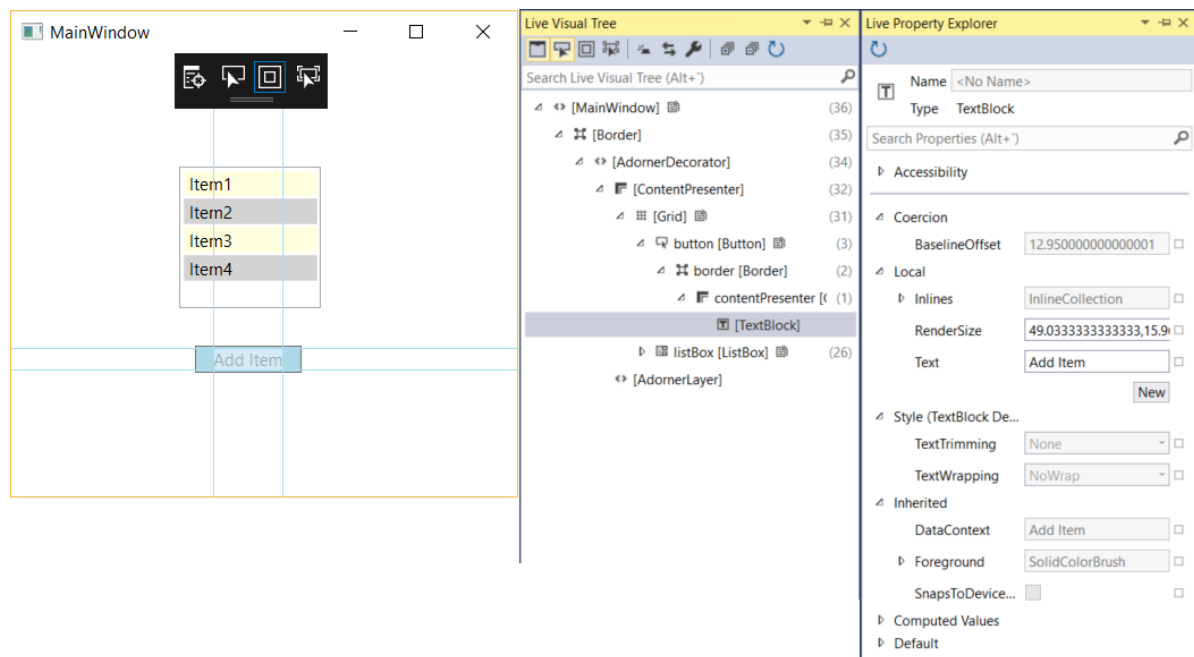
실행 중인 애플리케이션에 레이아웃 표시기를 표시합니다. 선택 사용 단추 바로 오른쪽에 있는 단추를 선택하면 이 모드를 사용할 수 있습니다. 레이아웃 표시기 표시를 켜면 애플리케이션 창에 선택한 개체의 범위를 따라 가로 및 세로줄이 표시됩니다. 따라서 여백을 표시하는 사각형뿐만 아니라 어떤 항목이 정렬되는지 확인할 수 있습니다. 예를 들어 요소 선택 및 레이아웃 표시를 모두 설정하고 응용 프로그램에서 항목 텍스트 블록 추가를 선택합니다. 라이브 시각적 트리에 텍스트 블록 노드가 표시되고 라이브 속성 뷰어에 텍스트 블록 속성이 표시되어야 합니다. 또한 텍스트 블록의 범위에는 가로 및 세로줄이 표시되어야 합니다.



선택 미리 보기. 라이브 시각적 트리 도구 모음의 왼쪽에서 세 번째 단추를 선택하면 이 모드를 사용할 수 있습니다. 이 모드에서는 애플리케이션의 소스 코드에 액세스할 수 있는 경우 요소가 선언된 XAML을 보여줍니다. 요소 선택 및 선택 영역 미리 보기를 선택한 다음 테스트 응용 프로그램에서 단추를 선택합니다. Visual Studio에서 MainWindow.xaml 파일이 열리고 커서가 단추가 정의된 줄에 위치합니다.

실행 중인 애플리케이션에서 선택을 사용합니다. 라이브 시각적 트리 도구 모음의 가장 왼쪽에 있는 단추를 선택하면 이 모드를 사용할 수 있습니다. 이 모드를 켜면 애플리케이션에서 UI 요소를 선택할 수 있고 라이브 시각적 트리(및 라이브 속성 뷰어)가 이 요소 및 해당 속성에 해당하는 트리에 노드를 표시하도록 자동으로 업데이트됩니다.

실행 중인 애플리케이션에 레이아웃 표시기를 표시합니다. 선택 사용 단추 바로 오른쪽에 있는 단추를 선택하면 이 모드를 사용할 수 있습니다. 레이아웃 표시기 표시를 켜면 애플리케이션 창에 선택한 개체의 범위를 따라 가로 및 세로줄이 표시됩니다. 따라서 여백을 표시하는 사각형뿐만 아니라 어떤 항목이 정렬되는지 확인할 수 있습니다. 예를 들어 선택 사용 및 레이아웃 표시를 모두 켜고 애플리케이션에서 항목 추가 텍스트 블록을 선택합니다. 라이브 시각적 트리에 텍스트 블록 노드가 표시되고 라이브 속성 뷰어에 텍스트 블록 속성이 표시되어야 합니다. 또한 텍스트 블록의 범위에는 가로 및 세로줄이 표시되어야 합니다.



선택 미리 보기. 라이브 시각적 트리 도구 모음의 왼쪽에서 세 번째 단추를 선택하면 이 모드를 사용할 수 있습니다. 이 모드에서는 애플리케이션의 소스 코드에 액세스할 수 있는 경우 요소가 선언된 XAML을 보여줍니다. 선택 사용 및 선택 미리 보기를 선택한 다음, 테스트 애플리케이션에서 단추를 선택합니다. Visual Studio에서 MainWindow.xaml 파일이 열리고 커서가 단추가 정의된 줄에 위치합니다.

응용 프로그램을 실행 하는 XAML 도구 사용

소스 코드가 없는 경우 이러한 XAML 도구를 사용할 수 있습니다. 실행 중인 XAML 애플리케이션에 연결하면 해당 애플리케이션의 UI 요소에서도 라이브 시각적 트리를 사용할 수 있습니다. 다음은 이전에 사용한 동일한 WPF 테스트 애플리케이션을 사용하는 예제입니다.

1. 릴리스 구성에서 **TestXaml** 애플리케이션을 시작합니다. 디버그 구성으로 실행 중인 프로세스에는 연결할 수 없습니다.
2. Visual Studio의 두 번째 인스턴스를 열고 **디버그 > 프로세스에 연결**을 클릭합니다. 사용 가능한 프로세스 목록에서 **TestXaml.exe**를 찾고 **연결**을 클릭합니다.
3. 애플리케이션이 실행되기 시작합니다.
4. Visual Studio의 두 번째 인스턴스에서 **라이브 시각적 트리**(**디버그 > 창 > 라이브 시각적 트리**)를 엽니다. **TestXaml** UI 요소가 표시되어야 하고 애플리케이션을 직접 디버그하는 동안 했던 것처럼 해당 요소를 조작할 수 있어야 합니다.

참고 항목

XAML 핫 다시 로드를 사용 하여 실행 중인 XAML 코드 작성 및 디버그

Visual Studio에서 XAML 핫 다시 로드를 사용 하여 실행 중인 XAML 코드 작성 및 디버그

2020-05-08 • 9 minutes to read • [Edit Online](#)

XAML 핫 다시 로드를 사용 하면 앱이 실행 되는 동안 XAML 코드를 변경할 수 있으므로 WPF 또는 UWP 앱 UI (사용자 인터페이스)를 빌드할 수 있습니다. 핫 다시 로드는 Visual Studio와 Blend for Visual Studio에서 모두 사용할 수 있습니다. 이 기능을 사용 하면 실행 중인 응용 프로그램의 데이터 컨텍스트, 인증 상태 및 디자인 타임 동안 시뮬레이션 하기 어려운 기타 실제 복잡성의 이점을 활용 하여 XAML 코드를 점진적으로 빌드 및 테스트할 수 있습니다. XAML 핫 다시 로드 문제를 해결 하는 데 도움이 필요한 경우 [Xaml 핫 다시 로드 문제 해결](#) 을 대신 참조 하세요.

NOTE

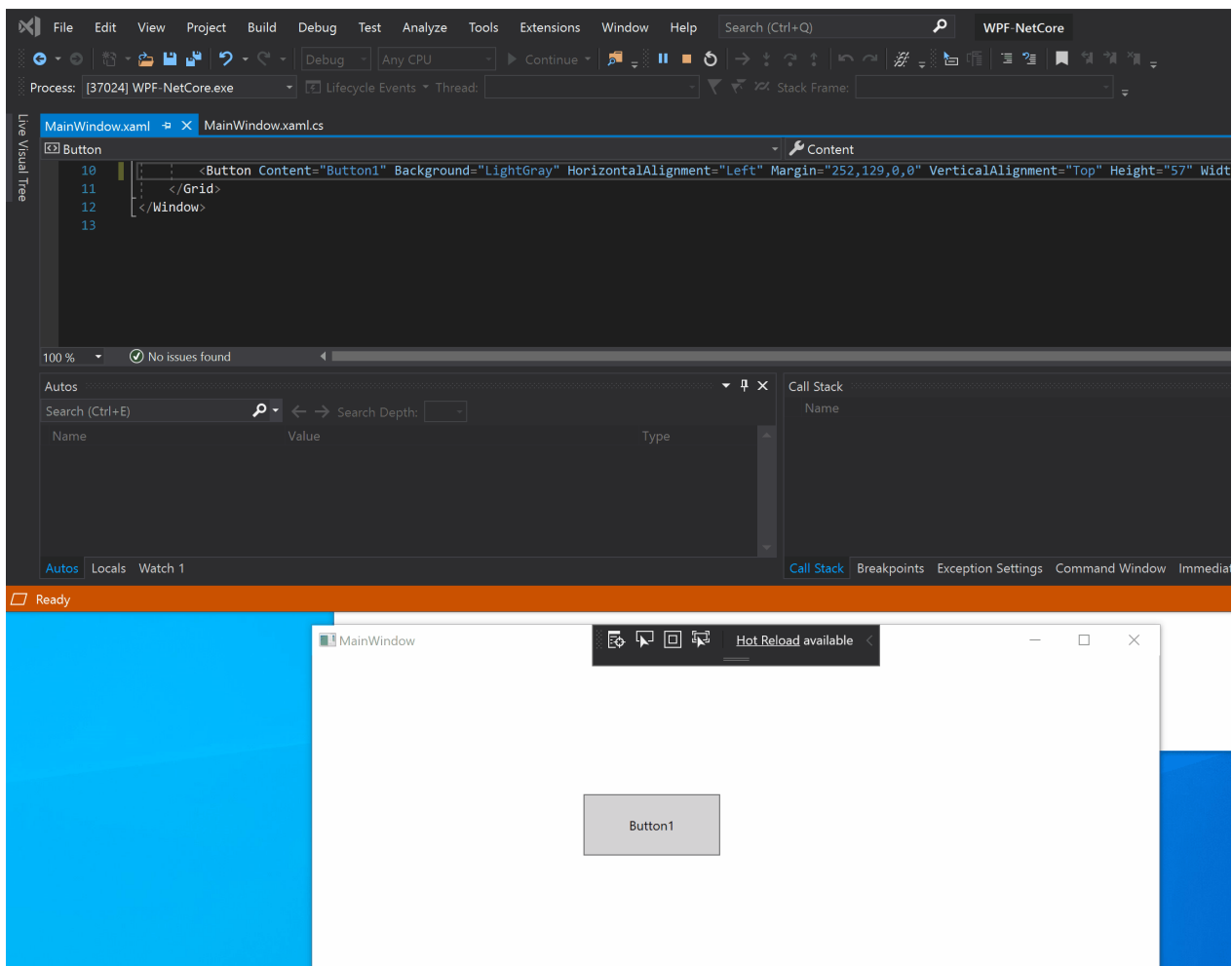
Xamarin.iOS를 사용 하는 경우 [xamarin.iOS에 대 한 XAML 핫 다시 로드](#)를 참조 하세요.

XAML 핫 다시 로드는 다음과 같은 시나리오에서 특히 유용 합니다.

- 응용 프로그램이 디버그 모드에서 시작 된 후 XAML 코드에서 발견 된 UI 문제를 수정 합니다.
- 응용 프로그램의 런타임 컨텍스트를 활용 하면서 개발 중인 앱에 대 한 새 UI 구성 요소를 구축 합니다.

지원 되는 응용 프로그램 유형	운영 체제 및 도구
WPF(Windows Presentation Foundation)	.NET Framework 4.6 + 및 .NET Core Windows 7 이상
UWP (유니버설 Windows 앱)	Windows 10 이상, windows 10 SDK 14393 +

다음 그림에서는 라이브 시각적 트리를 사용 하여 소스 코드를 연 다음 XAML 핫 다시 로드를 사용 하여 단추 텍스트 및 단추 색을 변경 하는 방법을 보여 줍니다.



NOTE

Visual Studio XAML 핫 다시 로드는 현재 Visual Studio에서 응용 프로그램을 실행 하거나 디버거가 연결 된 Blend for Visual Studio (F5 또는 디버깅 시작) 하는 경우에만 지원 됩니다. **환경 변수를 수동으로 설정** 하지 않는 한 **연결 프로그램 처리**를 사용 하 여이 환경을 사용 하도록 설정할 수 없습니다.

알려진 제한 사항

다음은 XAML 핫 다시 로드의 알려진 제한 사항입니다. 실행 되는 제한 사항을 해결 하려면 디버거를 중지 한 후 작업을 완료 합니다.

제한 사항	WPF	UWP	메모
앱이 실행 되는 동안 컨트롤에 이벤트 연결	지원되지 않음	지원되지 않음	오류: <i>이벤트 실패 확인</i> 을 참조 하세요. WPF에서는 기존 이벤트 처리기를 참조할 수 있습니다. UWP 앱에서 기존 이벤트 처리기를 참조 하는 것은 지원 되지 않습니다.

제한 사항	WPF	UWP	메 모
리소스 사전에 앱의 페이지/창 또는 <i>app.xaml</i> 과 같은 리소스 개체 만들기	Visual Studio 2019 업데이트 2부터 지원 됨	지원됨	예제:를으로 <code>SolidColorBrush</code> 사용할 리소스 사전에 추가 <code>StaticResource</code> 합니다. 참고: 정적 리소스, 스타일 변환기 및 리소스 사전에 작성 된 기타 요소는 XAML 핫 다시 로드를 사용 하는 동안 적용/사용할 수 있습니다. 리소스 만들기만 지원 되지 않습니다. 리소스 사전 <code>Source</code> 속성을 변경 합니다.
앱이 실행 되는 동안 프로젝트에 새 컨트롤, 클래스, 창 또는 기타 파일 추가	지원되지 않음	지원되지 않음	없음
NuGet 패키지 관리 (패키지 추가/제거/업데이트)	지원되지 않음	지원되지 않음	없음
{X:Bind} 태그 확장을 사용 하는 데이터 바인딩 변경	N/A	Visual Studio 2019부터 지원 됨	이 경우 Windows 10 버전 1809 (build 10.0.17763)이 필요 합니다. Visual Studio 2017 또는 이전 버전에서는 지원 되지 않습니다.
X:Uid 지시문 변경은 지원 되지 않습니다.	해당 없음	지원되지 않음	없음
다중 프로세스	지원되지 않음	지원되지 않음	핫 다시 로드는 한 번에 하나의 프로세스에 대해서만 사용할 수 있습니다.

오류 메시지

XAML 핫 다시 로드를 사용 하는 동안 다음 오류가 발생할 수 있습니다.

오류 메시지	DESCRIPTION
이벤트 실패 확인	오류는 응용 프로그램이 실행 되는 동안 지원 되지 않는 컨트롤 중 하나에 이벤트를 연결 하려고 시도 하고 있음을 나타냅니다.
이 변경은 XAML 핫 다시 로드에서 지원 되지 않으며 디버깅 세션 중에는 적용 되지 않습니다.	오류는 XAML 핫 다시 로드에서 시도 중인 변경을 지원 하지 않음을 나타냅니다. 디버깅 세션을 중지 하고, 변경 하고 나서, 디버깅 세션을 다시 시작 합니다. 지원 되지 않는 시나리오를 발견 한 경우 Visual Studio 개발자 커뮤니티 에서 새로운 "기능 제안" 옵션을 사용 하세요.

참고 항목

- [XAML 핫 다시 로드 문제 해결](#)
- [Xamarin에 대 한 XAML 핫 다시 로드](#)
- [편집하며 계속하기\(Visual C#\)](#)

XAML 핫 다시 로드 문제 해결

2020-05-08 • 7 minutes to read • [Edit Online](#)

이 문제 해결 가이드에는 XAML 핫 재로드가 제대로 작동 하지 못하게 하는 대부분의 문제를 해결 해야 하는 자세한 지침이 포함되어 있습니다.

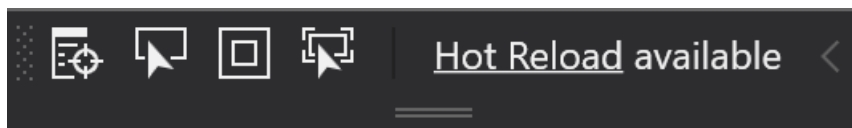
WPF 및 UWP 앱에는 XAML 핫 다시 로드가 지원 됩니다. 운영 체제 및 도구 요구 사항에 대 한 자세한 내용은 [Xaml 핫 다시 로드를 사용하여 실행 중인 xaml 코드 작성 및 디버그](#)를 참조 하세요.

핫 다시 로드를 사용할 수 없습니다.

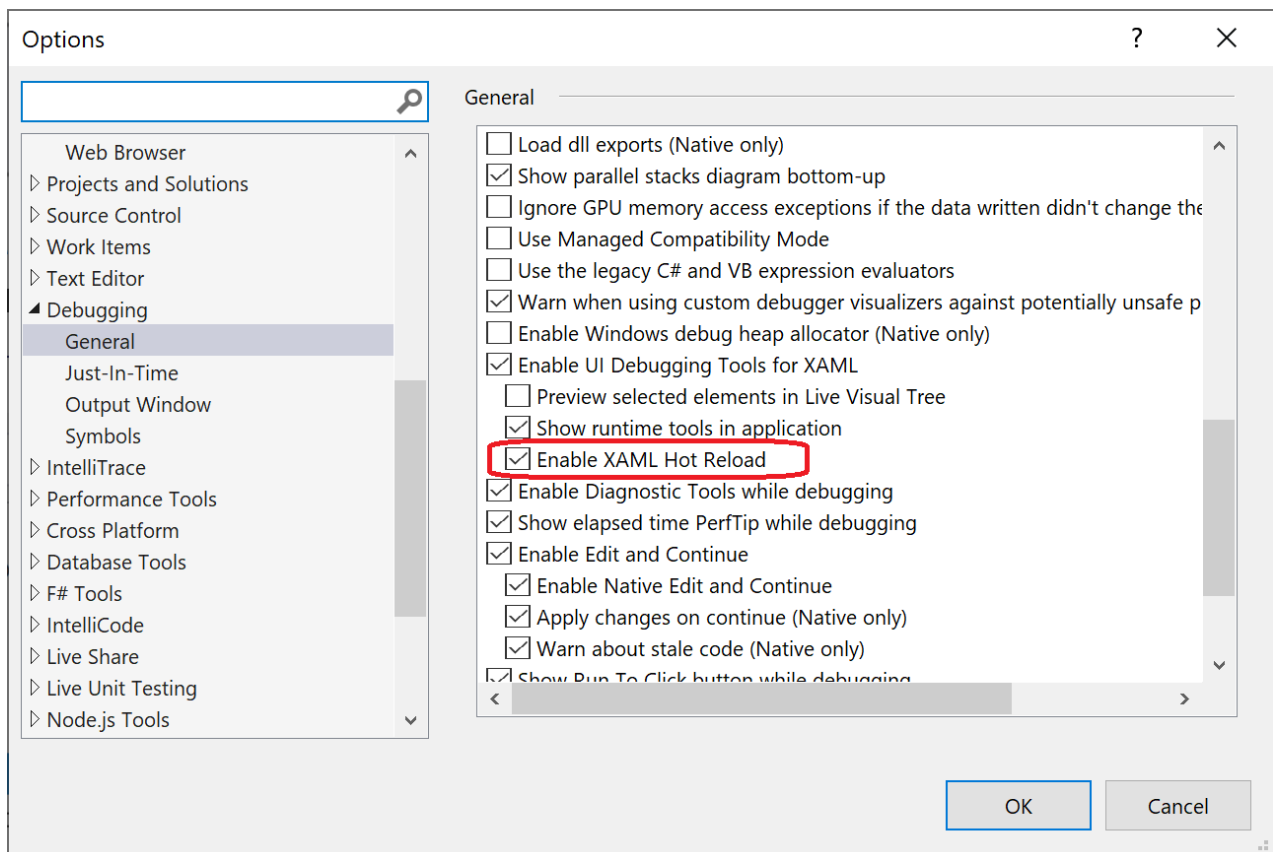
앱을 디버그 하는 동안 앱 내 도구 모음에서 "핫 다시 로드를 사용할 수 없습니다." 라는 메시지가 표시 되 면이 문서에 설명 된 지침에 따라 문제를 해결 합니다.

XAML 핫 다시 로드를 사용할 수 있는지 확인

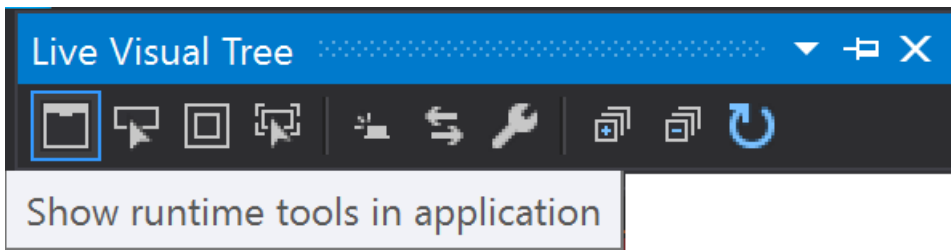
이 기능은 기본적으로 사용 하도록 설정 되어 있습니다. 앱 디버깅을 시작할 때 XAML 핫 다시 로드를 사용할 수 있는지 확인 하는 앱 내 도구 모음이 표시 되는지 확인 합니다.



앱 내 도구 모음이 표시 되지 않으면 **일반 디버그 > 옵션 >** 을 엽니다. 두 옵션, **xaml 용 UI 디버깅 도구 사용** 및 **Xaml 핫 다시 로드 사용** 이 선택 되어 있는지 확인 합니다.



이러한 옵션을 선택한 경우 라이브 시각적 트리 (디버그 > **Windows** > 라이브 시각적 트리)로 이동 하고 응용 프로그램 도구 모음에 런타임 도구 표시 단추 (맨 왼쪽)를 선택 해야 합니다.



프로세스에 연결 하는 대신 디버깅 시작을 사용 하는지 확인 합니다.

XAML 핫 다시 로드는 응용 프로그램이 시작 `ENABLE_XAML_DIAGNOSTICS_SOURCE_INFO` 될 때 환경 변수를 1로 설정 해야 합니다. Visual Studio는 디버그 > 디버깅 시작 또는 F5명령의 일부로이를 자동으로 설정 합니다. 디버그 > 프로세스에 연결 명령을 대신 사용 하여 XAML 핫 다시 로드를 사용 하려면 환경 변수를 직접 설정 합니다.

NOTE

환경 변수를 설정 하려면 시작 단추를 사용 하여 "환경 변수"를 검색 하고 시스템 환경 변수 편집을 선택 합니다. 열리는 대화 상자에서 환경 변수를 선택한 다음 사용자 변수로 추가 하고 값을로 1 설정 합니다. 정리 하려면 디버깅을 마친 후 변수를 제거 합니다.

MSBuild 속성이 올바른지 확인

기본적으로 소스 정보는 디버그 구성에 포함 됩니다. 프로젝트 파일 (예: *.csproj)의 MSBuild 속성에 의해 제어 됩니다. WPF의 경우 속성 `XamlDebuggingInformation` 은 이며로 `True` 설정 되어야 합니다. UWP의 경우 속성 `DisableXbfLineInfo` 은 이며로 `False` 설정 되어야 합니다. 다음은 그 예입니다.

WPF:

```
<XamlDebuggingInformation>True</XamlDebuggingInformation>
```

UWP

```
<DisableXbfLineInfo>False</DisableXbfLineInfo>
```

올바른 빌드 구성 이름을 사용 하고 있는지 확인 하십시오.

XAML 핫 다시 로드 (이전 섹션 참조)를 지원 하도록 올바른 MSBuild 속성을 수동으로 설정 하거나 기본 빌드 구성 이름 (디버그)을 사용 해야 합니다. MSBuild 속성을 올바르게 설정 하지 않으면 사용자 지정 빌드 구성 이름이 작동 하지 않고 릴리스 빌드가 되지 않습니다.

XAML 파일에 오류가 없는지 확인 합니다.

XAML 파일에 오류 목록오류가 표시 되 면 Xaml 핫 다시 로드가 작동 하지 않을 수 있습니다.

참고 항목

[XAML 핫 다시 로드를 사용 하여 실행 중인 XAML 코드 작성 및 디버그](#)

Blend에서 XAML 디버그

2020-05-08 • 14 minutes to read • [Edit Online](#)

Blend for Visual Studio의 도구를 사용 하여 앱에서 XAML을 디버그할 수 있습니다. 프로젝트를 빌드할 때 모든 오류는 **결과** 패널에 표시됩니다. 오류를 두 번 클릭하여 오류와 관련된 태그를 찾습니다. 작업할 공간이 더 필요한 경우 **F12**키를 눌러 **결과** 패널을 숨길 수 있습니다.

구문 오류

구문 오류는 XAML 또는 코드 숨김 파일이 언어의 서식 설정 규칙을 준수하지 않는 경우에 발생합니다. 오류 설명을 보면 오류 해결 방법을 파악할 수 있습니다. 오류가 발생하는 파일의 이름 및 줄 번호도 함께 설명됩니다. XAML 오류는 **결과** 패널의 **태그** 탭에 나열되어 있습니다.

TIP

XAML은 XML을 기반으로 하는 태그 언어로 XML 구문 규칙을 따릅니다.

XAML 구문 오류의 몇 가지 일반적인 원인은 다음과 같습니다.

- 키워드에 맞춤법 오류가 있거나 대문자 표시가 잘못되었습니다.
- 특성 또는 텍스트 문자열 주위에 따옴표가 없습니다.
- XAML 요소에 닫는 태그가 없습니다.
- XAML 요소가 허용되지 않는 위치에 있습니다.

공용 XAML 구문에 대한 자세한 내용은 [기본 XAML 구문 가이드](#)를 참조하십시오.

Blend에서 간단한 코드 숨겨진 구문 오류, 컴파일 오류 및 런타임 오류를 식별 하고 해결할 수도 있습니다. 하지만 코드 숨김 오류는 Visual Studio에서 더욱 쉽게 식별 및 해결할 수 있습니다.

디버깅 샘플 XAML 코드

다음 예제에서는 Blend에서 간단한 XAML 디버깅 세션을 안내 합니다.

프로젝트를 만들려면

1. Blend에서 **파일** 메뉴를 열고 **새 프로젝트**를 클릭 합니다.

새 프로젝트 대화 상자의 왼쪽에 프로젝트 형식 목록이 표시됩니다. 프로젝트 형식을 클릭하면 해당 프로젝트와 연결된 프로젝트 템플릿이 오른쪽에 표시됩니다.

2. 프로젝트 형식 목록에서 **Windows 유니버설**을 클릭 합니다.
3. 프로젝트 템플릿 목록에서 **비어 있는 앱 (유니버설 Windows)**을 클릭 합니다.
4. 이름 텍스트 상자에 입력 `DebuggingSample` 합니다.
5. 위치 텍스트 상자에서 프로젝트 위치를 확인합니다.
6. 언어 목록에서 **Visual C#**을 클릭한 다음, **확인**을 클릭하여 프로젝트를 만듭니다.
7. 디자인 화면을 마우스 오른쪽 단추로 클릭한 다음, **원본 보기**를 클릭하여 **분할** 보기로 전환합니다.
8. 코드의 오른쪽 상단에 있는 **복사** 링크를 클릭하여 다음 코드를 복사합니다.


```
<Grid HorizontalAlignment="Left" Height="222" VerticalAlignment="Top">
    <Button content="Button" x:Mame="Home" HorizontalAlignment="Left" VerticalAlignment="Top"/>
    <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="0,38,0,0">
    <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="0,75,0,0"/>
    <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="0,112,0,0"/>
    <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="0,149,0,0"/>
</Grid>
```

9. 기본 그리드를 찾아 여는 그리드 태그와 닫는 태그 사이에 코드를 붙여 넣습니다. 작업을 마치면 코드가 다음과 같이 됩니다.

```
<Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">
    <Grid HorizontalAlignment="Left" Height="222" VerticalAlignment="Top">
        <Button content="Button" x:Mame="Home" HorizontalAlignment="Left" VerticalAlignment="Top"/>
        <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top"
Margin="0,38,0,0">
        <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top"
Margin="0,75,0,0"/>
        <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top"
Margin="0,112,0,0"/>
        <Button Content="Button" HorizontalAlignment="Left" VerticalAlignment="Top"
Margin="0,149,0,0"/>
    </Grid>
</Grid>
```

10. Ctrl+Shift+B 를 눌러 프로젝트를 빌드합니다.

프로젝트를 빌드할 수 없음을 경고하는 오류 메시지가 표시되고, 오류를 나열하는 **결과** 패널이 앱 맨 아래에 나타납니다.

설명	파일	줄은선형	열	프로젝트
특성에서 "<" 값이 잘못되었습니다.	MainPage.xaml	12	15	디버깅...
공백이 없습니다.	MainPage.xaml	12	15	디버깅...
"0"은 이름의 시작 부분에 사용할 수 없습니다.	MainPage.xaml	16	98	디버깅...
공백이 없습니다.	MainPage.xaml	16	98	디버깅...
등호("=")가 없습니다.	MainPage.xaml	16	98	디버깅...
","는 이름에 사용할 수 없습니다.	MainPage.xaml	16	99	디버깅...
"1"은 이름의 첫 글자로 사용할 수 없습니다.	MainPage.xaml	16	100	디버깅...
등호("=")가 없습니다.	MainPage.xaml	16	100	디버깅...
","는 이름에 사용할 수 없습니다.	MainPage.xaml	16	103	디버깅...

XAML 오류 해결

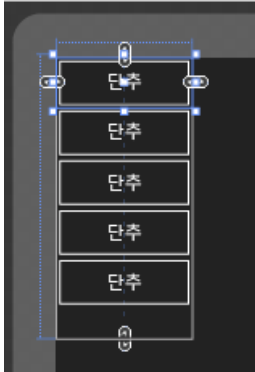
XAML 오류가 감지되면 디자인 화면에서 프로젝트에 잘못된 태그가 포함되어 있다는 경고를 표시합니다. 오류를 해결할 때 **결과** 패널의 오류 목록이 업데이트됩니다. 모든 오류를 해결하면 디자인 화면을 사용할 수 있게 되고 앱이 디자인 화면에 표시됩니다.

XAML 오류를 해결하려면

- 목록에서 첫 번째 오류를 두 번 클릭합니다. 설명은 "값 ' < '은 (는) 특성에서 사용할 수 없습니다."입니다. 오류를 두 번 클릭하면 포인터가 코드에서 해당하는 위치를 찾습니다. < 앞의 `Button` 은(는) 유효하지만 오류 메시지에서 제시한 대로 특성은 아닙니다. 코드의 이전 줄을 살펴보면 특성 `Top` 의 닫는 따옴표가 누락되어 있습니다. 닫는 따옴표를 입력합니다. **결과** 패널의 오류는 변경 내용을 반영하여 업데이트됩니다.
- " " 0 '은 (는) 이름 시작 부분에서 사용할 수 없습니다." 라는 설명을 두 번 클릭 합니다. `Margin="0,149,0,0"` 올바른 형식으로 표시 됩니다. 그러나 `Margin` 의 색 코딩이 해당 코드에서 `Margin` 의 다른 인스턴스와 일치하지 않습니다. 이전 이름/값 쌍(`VerticalAlignment="Top"`)에 닫는 따옴표가 누락되어 있기 때문에, `Margin="` 은 이전 특성 값의 일부분으로 읽고 0은 이름/값 쌍의 시작으로 읽습니다. `Top` 의 닫는 따옴표를 입력합니

다. 결과 패널의 오류 목록은 변경 내용을 반영하도록 업데이트됩니다.

3. 나머지 오류 "닫는 XML 태그 '단추'가 일치하지 않습니다."를 두 번 클릭합니다. 포인터는 닫는 **그리드** 태그 (`</Grid>`)에 있으므로 오류가 **Grid** 개체 안쪽에 있음을 나타냅니다. 두 번째 **Button** 개체에 닫는 태그가 누락되어 있습니다. 닫는 `/`를 추가한 후 **결과** 패널 목록이 업데이트됩니다. 이러한 초기 오류를 해결했으므로 두 가지의 추가 오류가 식별되었습니다.
4. "멤버 '콘텐츠'를 인식할 수 없거나 액세스할 수 없습니다."를 두 번 클릭합니다. `c`의 `content`는 대문자여야 합니다. 소문자 "c"를 대문자 "C"로 바꿉니다.
5. "'Mame' 속성이 `http://schemas.microsoft.com/winfx/2006/xaml` 네임 스페이스에 없습니다."를 두 번 클릭합니다. "Mame"의 "M"은 "N"이어야 합니다. "M"을 "N"으로 바꿉니다. XAML을 구문 분석할 수 있으므로 응용 프로그램이 디자인 화면에 나타납니다.



Ctrl+Shift+B를 눌러 프로젝트를 빌드하고 나머지 오류가 없는지 확인합니다.

Visual Studio에서 디버그

Visual Studio에서 Blend 프로젝트를 열어 앱에서 코드를 보다 쉽게 디버그할 수 있습니다. Visual Studio에서 Blend 프로젝트를 열려면 **프로젝트 패널**에서 프로젝트를 마우스 오른쪽 단추로 클릭한 다음 **Visual studio에서 편집**을 클릭합니다. Visual Studio에서 디버깅 세션을 마친 후 Ctrl + Shift + S를 눌러 변경 내용을 모두 저장한 다음 Blend로 다시 전환합니다. 프로젝트를 다시 로드할 것인지 묻는 메시지가 나타납니다. **모두 예**를 클릭하여 Blend에서 작업을 계속 진행합니다.

앱을 디버깅하는 방법에 대한 자세한 내용은 [Visual Studio에서 UWP 앱 디버그](#)를 참조하세요.

도움말 보기

Blend 앱을 디버깅하는 데 도움이 필요한 경우 [UWP 앱 커뮤니티 포럼](#)에서 문제와 관련된 게시물을 검색하거나 질문을 게시할 수 있습니다.

Visual Studio에서 유니버설 Windows 앱(UWP) 디버그

2020-05-18 • 2 minutes to read • [Edit Online](#)

Visual Studio 디버거는 UWP 앱 디버깅을 완벽하게 지원합니다. UWP 디버그 작업은 다음 문서를 참조하세요.

디버그 작업	기사
로컬 컴퓨터, 원격 또는 연결된 디바이스 또는 앱 시뮬레이터에서 UWP 앱을 실행합니다.	UWP 앱에 대한 디버깅 세션 시작
로컬 및 원격 모두에서 배포된 앱을 디버그합니다.	설치된 앱 패키지 디버그
XAML 코드 작성 및 실행 중인 XAML 코드 디버그	XAML 핫 다시 로드를 사용하여 UWP XAML 개체의 시각적 트리를 탐색하고 트리에 포함된 개체의 종속성 속성을 볼 수 있습니다. 이 항목에서는 XAML 핫 다시 로드를 사용하는 방법을 설명합니다.
디버그하는 동안 XAML 속성 검사	라이브 시각적 트리를 사용하여 UWP XAML 개체의 시각적 트리를 탐색하고 트리에 포함된 개체의 종속성 속성을 볼 수 있습니다.
UWP 앱에서 ContentPrefetcher 클래스의 효과를 분석합니다.	UWP 앱에 대한 콘텐츠 사전 인출

참조

- [정적 코드 분석을 사용하여 코드 품질 분석](#)

WPF 시작

2020-03-19 • 3 minutes to read • [Edit Online](#)

WPF(Windows Presentation Foundation)는 데스크톱 클라이언트 애플리케이션을 만드는 UI 프레임워크입니다. WPF 개발 플랫폼은 애플리케이션 모델, 리소스, 컨트롤, 그래픽, 레이아웃, 데이터 바인딩, 문서 및 보안을 포함하여 다양한 애플리케이션 개발 기능 집합을 지원합니다. 프레임워크는 .NET의 일부이므로 이전에 ASP.NET 또는 Windows Forms를 사용하여 .NET으로 애플리케이션을 빌드한 경우 프로그래밍 환경이 비슷합니다. WPF는 XAML(Extensible Application Markup Language)을 사용하여 애플리케이션 프로그래밍을 위한 선언적 모델을 제공합니다.

이 섹션의 항목에서는 WPF를 소개하고 WPF를 시작하는 데 유용한 정보를 제공합니다.

어디서 시작해야 합니까?

제목	아티클
바로 시작...	연습: 내 첫 WPF 데스크톱 애플리케이션
XAML 디자인 도구 비교...	Visual Studio 및 Blend for Visual Studio에서 XAML 디자인
.NET을 처음 사용하세요?	.NET 가이드 애플리케이션 주요 사항 Visual C# 시작
WPF에 대한 자세한 설명...	WPF 개요 XAML 개요(WPF) 컨트롤 데이터 바인딩 개요 LINQ to XML로 WPF 데이터 바인딩
Windows Forms 개발자인가요?	Windows Forms 컨트롤 및 해당 WPF 컨트롤 WPF 및 Windows Forms 상호 운용성에서 지원되는 시나리오

참고 항목

- [WPF용 데스크톱 가이드](#)
- [클래스 라이브러리\(WPF\)](#)
- [WPF 커뮤니티 리소스](#)
- [앱 개발 지원](#)

WPF 디버깅

2020-05-19 • 2 minutes to read • [Edit Online](#)

Visual Studio에서는 WPF 애플리케이션을 보다 쉽게 디버깅할 수 있는 추가 기능을 제공합니다.

관련 항목

제목	설명
XAML 코드 작성 및 실행 중인 XAML 코드 디버그	XAML 핫 다시 로드를 사용하여 WPF 개체의 시각적 트리를 탐색하고 트리에 포함된 개체의 WPF 종속성 속성을 볼 수 있습니다. 이 항목에서는 XAML 핫 다시 로드를 사용하는 방법을 설명합니다.
디버깅하는 동안 XAML 속성 검사	라이브 시각적 트리를 사용하여 WPF 개체의 시각적 트리를 탐색하고 트리에 포함된 개체의 WPF 종속성 속성을 볼 수 있습니다.
방법: WPF 트리 시각화 도우미 사용	WPF 트리 시각화 도우미를 사용하여 WPF 개체의 시각적 트리를 탐색하고 트리에 포함된 개체의 WPF 종속성 속성을 볼 수 있습니다. 이 항목에서는 WPF 트리 시각화 도우미의 사용자 인터페이스에 대해 설명합니다.
방법: WPF 추적 정보 표시	Visual Studio에서는 WPF 애플리케이션에서 디버그 추적 정보를 받아서 출력 창에 표시할 수 있습니다. 이 항목에서는 WPF 추적 정보를 표시하도록 설정하고 표시를 사용자 지정하는 방법을 보여 줍니다.

참조

- [관리 코드 디버그](#)

방법: WPF 트리 시각화 도우미 사용

2020-05-19 • 5 minutes to read • [Edit Online](#)

WPF 트리 시각화 도우미를 사용하여 WPF 개체의 표시 트리를 탐색하고 트리에 포함된 개체의 WPF 종속성 속성을 볼 수 있습니다. 시각적 트리에 관한 자세한 내용은 [WPF의 트리](#)를 참조하세요. 종속성 속성에 관한 자세한 내용은 [종속성 속성 개요](#)를 참조하세요.

WPF 트리 시각화 도우미를 열면 왼쪽에 **시각적 트리** 창이 표시되고 오른쪽에 '이름': '형식'의 속성 창이 표시됩니다. **시각적 트리** 창에서 개체를 선택하면 '이름': '형식'의 속성 창이 자동으로 업데이트되어 해당 개체의 속성이 표시됩니다.

NOTE

[라이브 시각적 트리 및 라이브 속성 탐색기](#)를 사용하여 WPF 개체의 시각적 트리를 살펴볼 수도 있습니다. WPF 트리 시각화 도우미는 레거시 기능이며 현재는 개발되지 않습니다.

WPF 트리 시각화 도우미를 열려면

1. DataTip, 조사식 창, 자동 창 또는 로컬 창에서 WPF 개체 이름 옆에 나타나는 돋보기 모양 아이콘 옆의 화살표를 클릭합니다.

시각화 도우미의 목록이 나타납니다.

2. WPF 트리 시각화 도우미를 클릭합니다.

표시 트리를 검색하려면

- 표시 트리 창의 검색 상자에 검색할 문자열을 입력합니다.

WPF 트리 시각화 도우미가 입력한 문자열과 일치하는 표시 트리의 첫 번째 개체를 즉시 찾습니다. 문자를 추가로 입력하여 좀 더 정확하게 일치하는 항목을 찾을 수 있습니다.

- 표시 트리 내에서 다음 일치 항목으로 이동하려면 **다음**을 클릭합니다.
- 이전 일치 항목으로 이동하려면 **이전**을 클릭합니다.
- 검색 조건을 지우려면 **지우기**를 클릭합니다.

속성 목록을 검색하려면

- '이름': '형식'의 속성 창의 필터 상자에 검색할 문자열을 입력합니다.

WPF 트리 시각화 도우미가 입력한 문자열과 일치하는 속성을 즉시 찾습니다. 이제 목록에는 입력한 문자열과 일치하는 속성만 표시됩니다. 문자를 추가로 입력하여 좀 더 정확하게 일치하는 항목을 찾을 수 있습니다.

- 검색 조건을 지우려면 **지우기**를 클릭합니다.

시각화 도우미를 닫으려면

- 대화 상자의 오른쪽 위 모퉁이에서 **닫기** 아이콘을 클릭합니다.

참조

- [Create Custom Visualizers of Data](#)(데이터의 사용자 지정 시각화 도우미 만들기)
- [WPF의 트리](#)
- [종속성 속성 개요](#)

방법: WPF 추적 정보 표시

2020-05-19 • 5 minutes to read • [Edit Online](#)

Visual Studio에서는 WPF 애플리케이션에서 디버그 추적 정보를 받아서 **출력** 창에 표시할 수 있습니다. 디버그 추적 정보를 표시하려면 WPF 추적을 사용하도록 설정해야 합니다.

WPF 추적 설정에는 App.Config 파일을 사용하거나 [PresentationTraceSources](#) 클래스를 통한 프로그래밍 방식을 사용할 수 있습니다. 더 쉽게 WPF 추적을 사용하도록 설정하는 방법은 **옵션** 창을 사용하는 것입니다. 애플리케이션에 대한 WPF 추적은 지원되지 않습니다.

WPF 추적 정보를 사용하도록 설정하거나 사용자 지정하려면

1. 도구 메뉴에서 **옵션**을 선택합니다.
2. 옵션 대화 상자의 왼쪽에 있는 상자에서 **디버깅** 노드를 엽니다.
3. 디버깅에서 **출력** 창을 클릭합니다.
4. 일반 출력 설정에서 모든 디버그 출력을 선택합니다.
5. 오른쪽에 있는 상자에서 **WPF 추적 설정**을 찾습니다.
6. **WPF 추적 설정** 노드를 엽니다.
7. **WPF 추적 설정**에서 사용하도록 설정할 설정의 범주(예: 데이터 바인딩)를 클릭합니다.
드롭다운 목록 컨트롤이 데이터 바인딩 또는 클릭한 범주 옆의 설정 열에 나타납니다.
8. 드롭다운 목록을 클릭하고 다음과 같이 보려는 추적 정보 형식을 선택합니다. **All**, **Critical**, **Error**, **Warning**, **Information**, **Verbose** 또는 **ActivityTracing**.
Critical은 중요 이벤트만 추적하도록 설정합니다.
Error는 중요 및 오류 이벤트를 추적하도록 설정합니다.
Warning은 중요, 오류 및 경고 이벤트를 추적하도록 설정합니다.
Information은 중요, 오류, 경고 및 정보 이벤트를 추적하도록 설정합니다.
Verbose는 중요, 오류, 경고, 정보 및 자세한 이벤트를 추적하도록 설정합니다.
ActivityTracing은 중지, 시작, 일시 중단, 전송 및 다시 시작 이벤트를 추적하도록 설정합니다.
이러한 추적 정보 수준의 의미에 대한 자세한 내용은 [SourceLevels](#)를 참조하십시오.
9. 확인을 클릭합니다.

WPF 추적 정보를 사용하지 않도록 설정하려면

1. 도구 메뉴에서 **옵션**을 선택합니다.
2. 옵션 대화 상자의 왼쪽에 있는 상자에서 **디버깅** 노드를 엽니다.
3. 디버깅에서 **출력** 창을 클릭합니다.
4. 오른쪽에 있는 상자에서 **WPF 추적 설정**을 찾습니다.
5. **WPF 추적 설정** 노드를 엽니다.
6. **WPF 추적 설정**에서 사용하도록 설정할 설정의 범주(예: 데이터 바인딩)를 클릭합니다.

드롭다운 목록 컨트롤이 데이터 바인딩 또는 클릭한 범주 옆의 설정 열에 나타납니다.

7. 드롭다운 목록을 클릭하고 **해제**를 선택합니다.

8. **확인**을 클릭합니다.

참조

- [WPF 디버그](#)