Microsoft ROBOTICS STUDIO

Microsoft Visual Programming Language (VPL)

Visual Programming Langage는 아래의 항목으로 구성됩니다.

목차

1. 시작하기

VPL 소개

마이크로소프트 비주얼 프로그래밍 언어(VPL)는 이전의 프로그램 작성에서 보이는 제어 흐름보다 그래픽적 데이터 흐름에 기반을 둔 프로그래밍 모델로 디자인된 애플리케이션 개발 환경입니다. 데이터 플로우 프로그램은 명령의 집합을 순차적으로 실행하는 것이라기 보다는 재료가 도착하는 것에 따라 주어진 업무를 수행하는 조립 라인의 일련의 근로자와 같습니다. 그 결과 VPL 은 여러 가지의 동시형 또는 분산 처리 시나리오를 프로그램하는 것에 아주 적합합니다.

VPL 은 변수와 논리와 같은 개념에 대한 기본 이해가 필요한 초보 프로그래머에 적합합니다. 그러나, VPL 은 초보자에게 제한되지 않습니다. 프로그래밍 언어의 구성적 성질은 빠른 프로토타이핑 또는 코드 개발을 위해 고급 프로그래머에게도 매우 유용할 것입니다. 또한, VPL 의 툴박스가 로봇 애플리케이션을 개발에 맞춰져 있지만 근본적인 구조는 로봇 프로그래밍에 제한되지 않고, 다른 애플리케이션에도 적용될 수 있습니다. 그 결과, VPL 은 학생, 애호가/동호인을 포함해 웹 개발자와 전문 프로그래머까지 넓은 사용자들에게 흥미로울 것입니다.

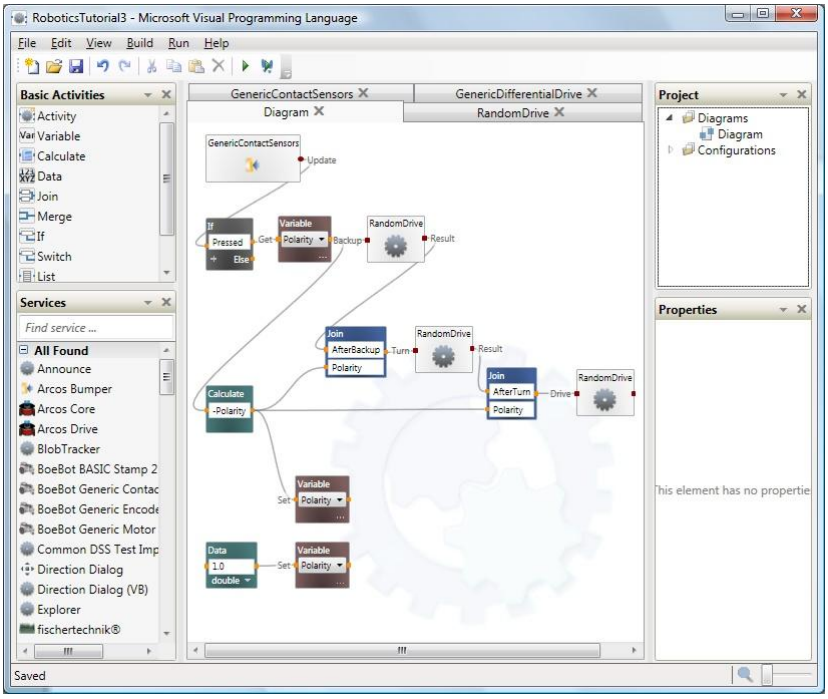


그림 1 – 단순한 bump-turn-go 탐색 행동을 위한 비주얼 프로그래밍 언어(VPL) 다이어그램

마이크로소프트 비주얼 프로그래밍 언어 데이터 플로우는 다른 activity 블록에 연결될 수 있는 입력과 출력을 가진 블록으로 표현된 activity 들의 연결된 순서로 구성됩니다.

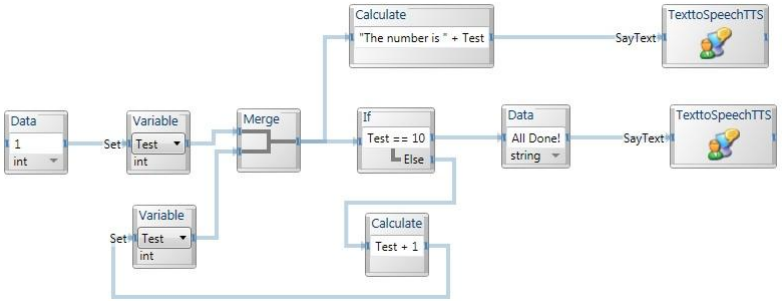
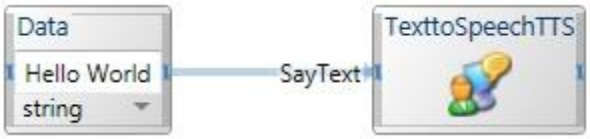


그림 2 – 한 개의 activity 에서 다른 것으로 데이터를 전송하는 메시지를 나타내는 블록 사이의 연결 화살표

Activitie 는 사전에 빌드된 서비스, 데이터 흐름 제어, 함수 또는 다른 코드 모듈을 표현할 수 있습니다. 결과적인 애플리케이션은 그러므로 개별 프로세스의 순서화인 오케스트레이션이라 불립니다.

Activity 는 또한 다른 activity 들의 조합을 포함할 수 있습니다. 이것은 activity 을 조합하고, 빌딩 블록으로 조합을 재사용하는 것을 가능하게 합니다. 이점에서 VPL 로 만들어진 애플리케이션은 그 자체로 activity 입니다.

Activity 블록은 그 연결 점을 나타내는 activity 의 이름과 경계를 일반적으로 포함합니다. activity 블록은 또한 사용자가 activity 에 사용될 데이터 값 입력, 할당 또는 변환을 위한 사용자 인터페이스 요소 및 activity 의 목적을 설명하기 위해 그래픽을 포함할 수 있습니다.

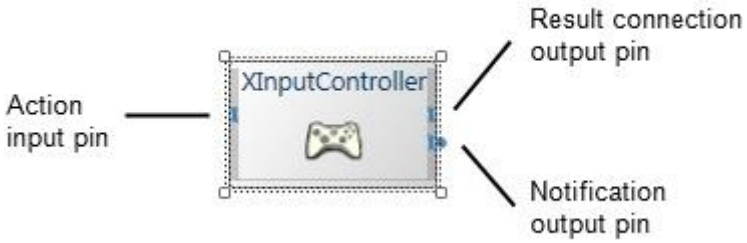


Activity 는 그들의 연결 핀을 통하여 연결 됩니다. activity 의 좌측의 연결 핀은 입력 메시지를 위해 연결 점을 표시합니다. 그리고 우측 핀은 출력 메시지를 위한 연결 점을 표시합니다.

Activity 는 그 입력 연결 핀을 통하여 데이터가 포함된 메시지를 수신합니다. activity 의 입력 핀은 action 또는 핸들러로 알려진 미리 정의된 내부 함수로의 연결 점입니다.

유효한 입력 메시지를 수신하면 activity 블록은 가동되고 수신된 메시지 데이터를 처리합니다. activity 에 보내지는 모든 데이터는 해당 activity 에 의해 소모됩니다. activity 의 출력을 통해 데이터를 포워드하기 위해서는 수신 activity 는 데이터를 복제해야 하고, 그것을 출력 연결에 넣어야 합니다.

Activity 는 다중 입력 연결 핀과 그 각각에 대한 출력 연결 핀을 가질 수 있습니다. 출력 연결 핀은 두 개의 종류입니다: 결과 출력 또는 알림 출력(때때로 이벤트 또는 publication 출력이라고 불림) 중에 하나일 수 있습니다. 결과 출력은 직사각형 연결 핀으로 publication 출력은 둥근 연결 핀으로 표시됩니다.



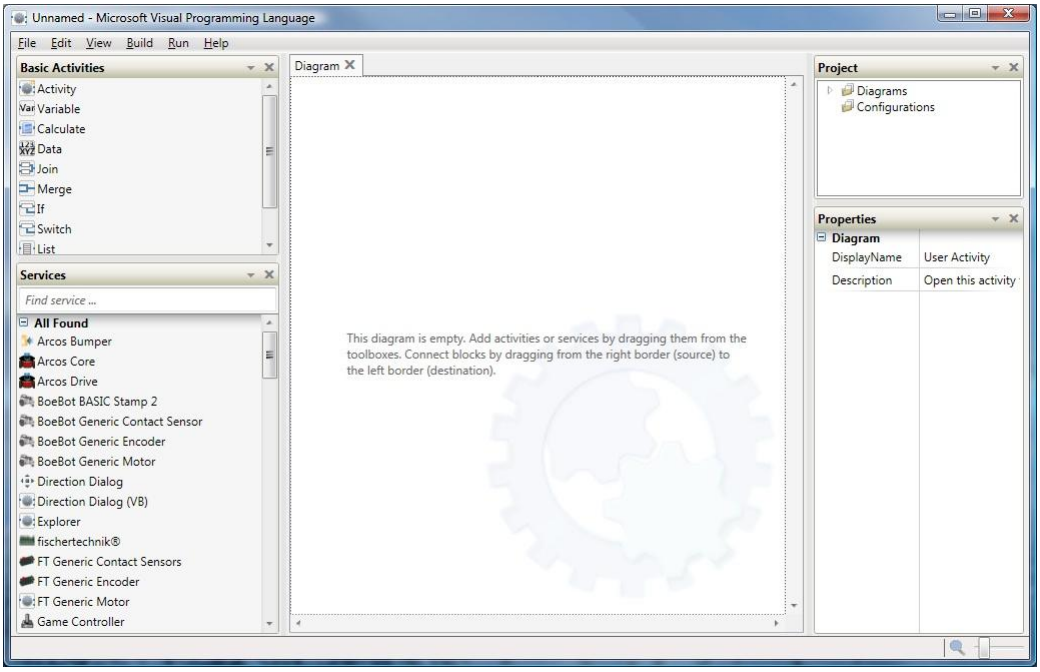
출력 메시지(데이터)가 특정 입력 실행 메시지의 결과로 전송될 때 응답 출력 핀이 사용됩니다. 알림 핀은 입력된 activity 메시지의 결과로 정보를 보내는 데 사용되지만, 일반적으로 내부 상태변화를 메시지로 전송할 때 사용됩니다. 결과 핀은 단지 입력 메시지의 수신에서 대해 하나의 메시지를 보내지만 알림 핀은 또한 메시지 여러 번 생성할 수 있습니다. 따라서 알림 출력 핀은 메시지 데이터를 반복적으로 전송할 때 사용됩니다.

시작메뉴의 "마이크로소프트 Robotics Studio" 아래에서 VPL 을 시작할 수 있습니다. VPL 다이어그램을 만드는 튜토리얼은 VPL Tutorials Overview 에 있습니다.

2. 사용자 가이드

다이어그램 생성

마이크로소프트 비주얼 프로그래밍 언어를 시작하면 다음과 같은 메뉴, 툴박스와 탭이 붙여진 다이어그램 페이지로 된 창을 볼 것입니다.



툴박스는 프로젝트의 현재의 파일 내용과 다이어그램을 만들기 위해 사용할 수 있는 activity 을 표시합니다. 툴박스는 이동할 수 있고 접을 수 있습니다. 여러분은 주요한 VPL 창 내에서 이들을 재배치할 수 있습니다.

View 메뉴에서 툴 박스 명령을 사용해 숨겨진 툴박스 창을 다시 표시할 수 있습니다.

데이터 플로 다이어그램을 시작으로 툴박스에서 페이지에 탭이 붙여진 다이어그램 영역으로 데이터 플로우 activity 또는 서비스 블록을 드래그 앤 드롭하고, 이들을 연결합니다. 또한 툴박스 엔트리를 더블 클릭하여 새로운 activity 을 추가할 수 있습니다. 툴박스 엔트리들에 커서를 움직이면 해당 activity 의 설명을 포함하는 툴팁(tooltip)이 나타납니다.

편집(Edit) 메뉴 또는 activity 팝업 문맥 메뉴에 있는 명령을 사용해 다이어그램에서 블록을 선택해 activity 를 자르거나(cut), 복사하거나(copy), 붙이거나(paste), 삭제(delete)할 수 있습니다.

기본 Activity 툴박스 창

기본 Activity 툴박스 창은 데이터 플로우를 제어하고, 데이터와 변수를 만들기 위한 블록을 포함합니다. 또한 이 툴박스는 텍스트 주석 블록을 여러분의 다이어그램에 배치하는 커맨트 Activity 을 포함합니다.

서비스 툴박스 창

서비스 툴박스는 VPL 과 호환 가능한 서비스를 표시합니다.

여러분의 다이어그램의 안에 이미 존재하는 서비스를 툴박스에서 드래그할 경우, 서비스의 새로운 인스턴스를 만들지 혹은 다이어그램에서 서비스에 참조를 생성할 지를 물어 봅니다.

툴박스의 최상위에 텍스트 박스에 새로운 이름을 입력해 해당 서비스의 이름을 변경할 수 있습니다.

툴박스 윈도우에 표시되는 서비스는 검색 필터(질의)박스의 컨텐츠에 의해 표시됩니다. 단어를 입력해 해당하는 서비스를 찾을 수 있습니다. 서비스 이름과 함께 서비스 설명에서도 입력된 단어로 서비스를 찾을 수 있습니다. 단어 사이에 +를 두어 양쪽 단어가 모두 만족하는 서비스를 찾거나 - 다음에 단어를 입력해 서비스 검색에서 제외할 수 있습니다.

찾아진 머릿글(heading)에서 +(add)버튼을 클릭하여 검색 필터를 저장할 수 있습니다. 만일 필터 단어 없이 검색 필터를 위한 타이틀을 만들고 싶다면, "= title"을 검색 텍스트에 추가하십시오. 모든 찾아진 머릿글에서 +버튼을 클릭하면 새로운 섹션은 단지 해당 텍스트를 사용할 것입니다. 예를 들어 다음을 입력하면:

```
NXT RCX = LEGO services
```

그리고 +버튼을 클릭하면 이름, 또는 설명에서 NXT RCX 단어가 포함된 서비스를 가진 LEGO Services 로 표제를 붙여진 새 섹션을 볼 수 있습니다.

프로젝트 창

프로젝트 창은 여러분의 프로젝트에 포함된 다이어그램과 설정 파일(서비스를 위해)을 보여줍니다.

다이어그램의 현재의 보기를 닫기 위해서 다이어그램의 탭에서 X 버튼을 클릭하십시오. 다이어그램을 다시 표시하기 위해 프로젝트 창에서 다이어그램의 이름을 더블 클릭하십시오. 만일 다이어그램 이름이 현재 표시되지 않으면 + 버튼을 다이어그램(Diagrams) 엔트리를 확장하기 위해 클릭하십시오.

다이어그램의 이름을 변경하기위해 프로젝트 창에서 해당 엔트리를 클릭하고 프로퍼티 뷰어에서 그 이름을 변경합니다.

프로젝트에 부가적인 다이어그램을 추가하기 위해 다이어그램 엔트리에 대한 문맥(context) 팝업 메뉴를 표시하고 Add Diagram 을 클릭합니다.

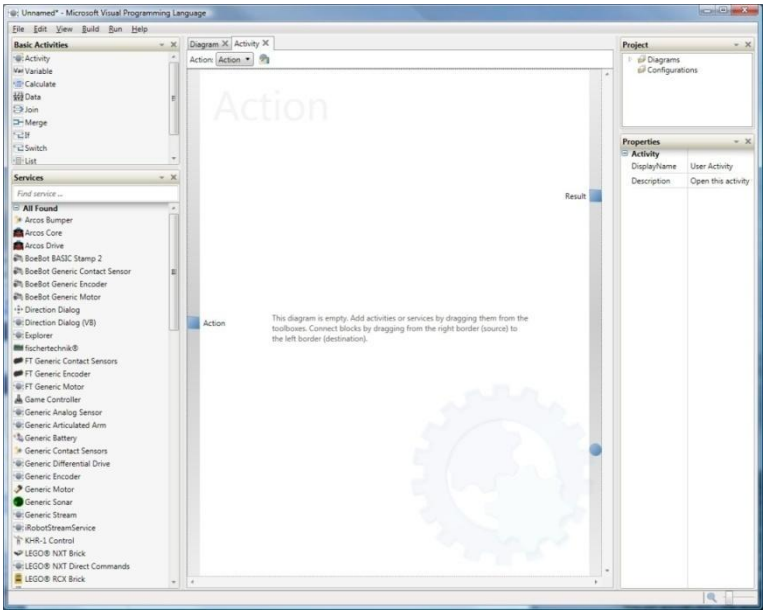
다이어그램을 삭제하기 위해 다이어그램에 대한 문맥 팝업 메뉴를 표시하고 Delete 를 클릭합니다.

프로퍼티 창

프로퍼티 창은 현재 선택된 아이템에 대한 프로퍼티를 표시합니다. 이것은 그 서비스 또는 Activity 에 대한 설정을 조절하게 합니다.

Activity 생성과 편집

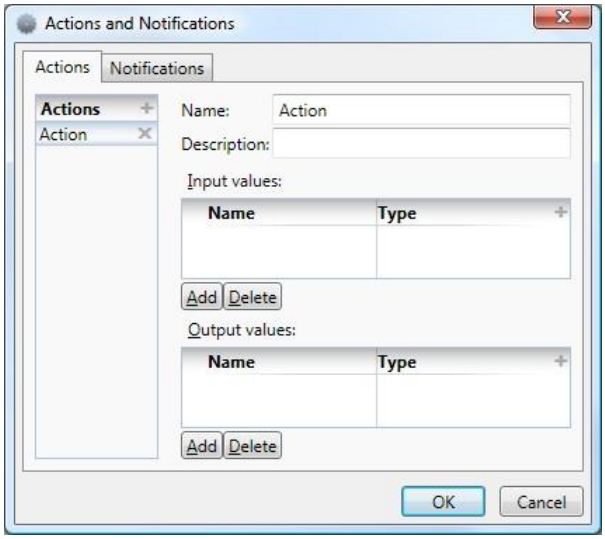
VPL 로 여러분의 Activity 를 만들 수 있습니다. 기본 activity 툴박스에서 activity 아이콘을 드래그하고 그것을 더블클릭 하거나 context 팝업 메뉴를 사용해 여십시오. 이것은 activity 에 대한 새로운 탭이 붙여진 다이어그램 창을 열 것입니다.



VPL 은 여러분을 위한 디폴트 Action 템플릿을 만듭니다. Activity 를 열 때 만일 Action 페이지가 표시되지 않으면, 탭 영역 아래의 Action dropdown 을 사용하십시오. 여러분이 메인 데이터 플로우 다이어그램에서 한 것 같이 Activity 를 삽입할 수 있고, 그들을 연결할 수 있습니다.

그러나, 다른 데이터 플로우가 여러분의 내부 데이터 플로우를 사용하게 하기 위해, 당신은 내부 데이터 플로우를 activity 의 외부 핀에 연결해야 합니다. 단순히 액션 핀에서 입력으로 연결을 드래그 하여 연결합니다.

Activity 의 외부 연결 핀을 위한 이름과 데이터타입을 정의하기 위해, Edit 메뉴에서 Actions and Notifications 명령을 선택하십시오. 이름 또는 타입을 변경하기 위해 목록에서 엔트리를 선택하여 변경합니다.



만일 여러 번 출력하는 activity 의 데이터 플로우라면 그 출력을 알림(Notification) 연결로 설정하십시오. Notifications 탭을 클릭하여 Actions 와 Notifications 다이얼로그를 사용할 수 있습니다. 그리고 Add 버튼을 클릭하여 이름과 타입을 지정합니다. OK 를 클릭하고 난 후에 여러분의 데이터 플로우 출력에서 방금 생성된 둥근 알림 연결로 드래그 할 수 있습니다.

Actions 와 Notifications 다이얼로그로 여러분의 activity 를 위한 action 핸들러를 만들 수 있습니다. Action 핸들러를 추가하기 위해, 이 다이얼로그에서 Actions 탭을 클릭하십시오 그리고 Add 버튼을 클릭하십시오. 새롭게 만들진 action 으로 전환하기 위해 Activity 페이지의 최상위에 Action dropdown 에서 그것을 선택하십시오.

때때로 여러분은 입력 또는 출력 연결을 필요하지 않는 activity 을 만들지도 모릅니다. 이런 상황을 위해 Start action 을 사용할 수 있습니다. VPL 은 여러분의 activity 를 위해 Start action 페이지를 자동으로 포함합니다. 연결됨에 관계없이 여러분의 activity 를 시작할 때 실행하고 싶어하는 데이터 플로우에 이것을 사용할 수 있습니다.

서비스 설정

서비스는 시작되는(초기 상태 설정) 방법과 필요한 "파트너 서비스"를 지정하기 위해 설정 정보가 필요합니다. 이 정보는 서비스가 선택된 상태에서 프로퍼티 창에 표시 되거나 Edit 메뉴나 context 팝업 메뉴에서 Set Configuration 명령을 클릭하거나 서비스 블록을 더블 클릭하여 Set Configuration 페이지를 열어 확인 가능합니다.

서비스에 따라 서비스의 초기 설정을 위한 네 개의 옵션 중의 하나를 선택할 수 있습니다.

초기 설정 지정하기

이 옵션에서는 서비스(인스턴스)의 초기 상태 값을 설정하고 필요한 파트너 서비스를 선택할 수 있습니다. 파트너 설정을 위해 서비스 정의를 사용하거나 사용할 다른 서비스를 선택할 수 있습니다.

서비스는 설정을 포함하는 상태를 가질 수 있습니다. 그 초기 상태를 설정하여 서비스를 구성합니다. 설정은 시리얼 통신 포트와 보레이트와 같은 상태 프로퍼티 뿐만 아니라 드라이브 서비스를 위한 바퀴 사이 거리와 바퀴 직경과 같은 물리적 프로퍼티를 포함합니다. 이러한 초기 설정 프로퍼티는 프로퍼티 창이나 Set Configuration 페이지의 Initial State 탭에서 지정할 수 있습니다.

일부 서비스는 그들의 조작을 위해 사용하는 파트너를 정의합니다. 예를 들면 디퍼런셜 드라이브 서비스는 두 개의 모터 서비스를 필요로 합니다. 선택적으로 그것은 또한 두 개의 바퀴 엔코더를 사용할 수 있습니다. 여러분은 해당 파트너로 사용할 서비스를 지정하기 위해 툴박스에서 Set Configuration 페이지의 Partners 탭의 엔트리로 드래그하거나 프로퍼티 창의 드롭 다운 리스트에서 파트너를 선택할 수 있습니다. 파트너로 선택된 서비스 또한 설정될 수 있습니다.

서비스 초기 설정을 지정할 때 VPL 은 프로젝트에 이를 파일로 추가합니다. 초기 설정을 지정된 채로 서비스를 삭제할 경우 이를 확인하는 경고가 표시되고 설정 파일일 함께 지우게 됩니다. 또한 설정 파일 선택한 뒤 context 팝업 메뉴에서 Delete 를 클릭하여 삭제할 수 있습니다.

다른 서비스를 사용하기

이 옵션에서는 서비스 설정에 다른 서비스의 구현을 사용합니다. 사용할 특정 서비스를 직접 선택하거나 DSS 런타임이 적당한 서비스를 찾거나 생성하게 할 수 있습니다. 이것은 여러분의 다이어그램에서 제네릭 서비스를 사용하게 해주고, 다이어그램을 변경하지 않고 사용되는 실제 서비스를 간단히 변경하는 것을 가능하게 합니다. 선택한 서비스는 또한 구성될 수 있습니다.

매니페스트 사용하기

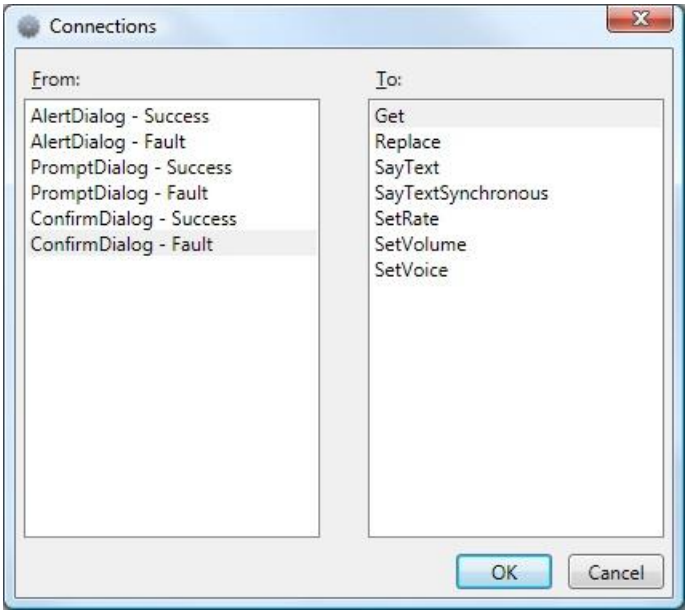
이 옵션은 서비스를 시작하기 위해 기존 매니페스트 파일을 선택해 사용합니다. 매니페스트는 시작될 서비스들과 이에 대한 설정을 기술하는 특수 파일입니다. Import Manifest 명령을 사용해 선택 가능한 기존 매니페스트의 목록을 표시합니다. 그 다음 서비스를 시작할 매니페스트를 선택할 수 있습니다. 또한 DSS 런타임이 적합한 서비스를 찾거나 생성하게 할 수 있습니다. 매니페스트를 만들기 위해서는 "로보틱스 튜토리얼 3"를 참고하십시오.

미지정(None)

이 옵션은 DSS 런타임이 적합한 서비스를 발견하거나 만들게 합니다. 이것은 디폴트 옵션입니다. 이 옵션은 대부분의 단순한 서비스(예를 들면 Math Functions, Simple Dialog 또는 Text-To-Speech 서비스)에 적당합니다.

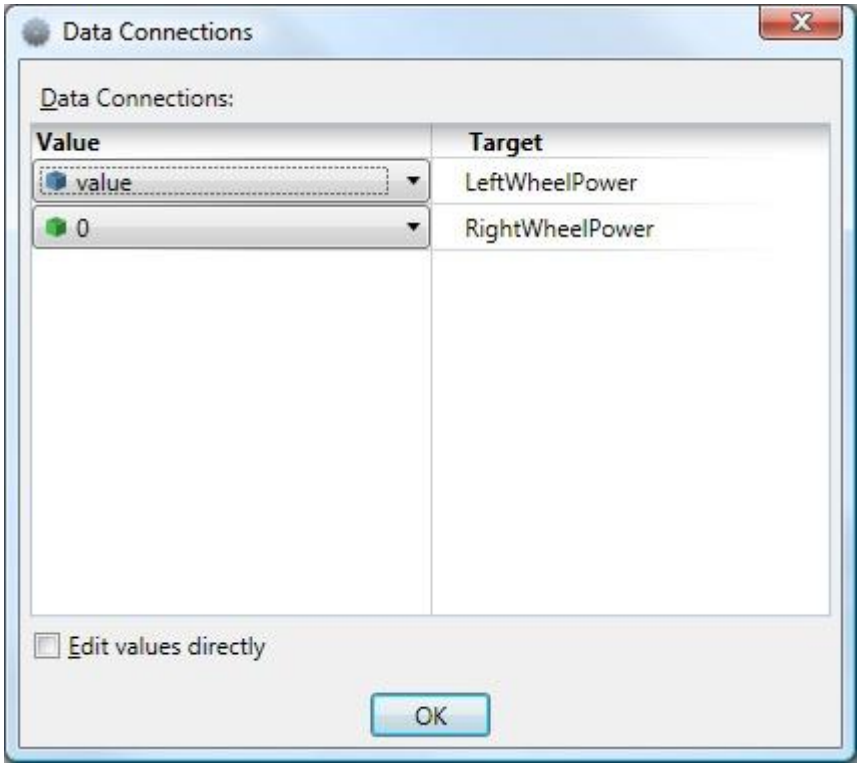
Activity 연결

두 개의 Activity 블록을 연결하기 위해, 한 개의 activity 연결 핀으로부터 다른 한편의 핀으로 드래그합니다. 만일 activity 를 연결하는 여러 방법(출력과 입력)이 있으면 다음과 같이 입력과 출력의 연결을 정의하는 Connections 대화 상자가 나타날 것입니다.



연결을 완료하기 위해 왼쪽 목록에서 결과 출력 연결을 선택합니다. 그리고 오른쪽 목록에서 action 을 선택한 후 OK 를 클릭합니다. 링크가 activity 사이에 나타날 것입니다. 만일 무연결(no connection) 옵션이 있으면, Connections 다이얼로그는 나타나지 않습니다. 종종 activity 의 출력에 이용 가능한 옵션은 그 입력 연결의 종류에 따라 결정됩니다.

때때로 Activity 사이의 연결에서 전달되는 데이터의 매칭이 요구됩니다. 이 경우 Data Connections 다이얼로그가 나타나고 메시지 전송자의 데이터 옵션이 좌측에 수신자(receiver)가 우측에 표시됩니다. 입력 데이터 옵션은 그 데이터 타입(수의 타입에 0, 불리안 타입에 false, 모든 다른 타입에 null), 데이터의 값 또는 서브 데이터(데이터 구성 요소)값에 대해 디폴트 값을 포함합니다.



여러분은 한 Activity 에서 다른 것으로 연결된 연결 링크를 선택해 원하는 다른 쪽으로 드래그하여 바꿀 수 있습니다. 연결 링크에서 팝업 context 메뉴를 선택해서 연결이나 데이터 조합을 바꿀 수 있습니다. 연결 링크는 링크를 선택하고 Del 키를 누르거나 팝업 context 메뉴 또는 Edit 메뉴에서 Delete 를 선택해 삭제할 수 있습니다.

만일 Activity 가 몇 개의 입력 연결(action 핸들러)을 지원할 경우는 한번에 한 개만 연결할 수 있습니다. 이미 activity 와 링크를 연결한 경우 다른 입력 핀 중 하나에 연결하고자 하면 Activity 블록의 카피(참조)해서 새로 만들고 이를 연결해서 사용하십시오.

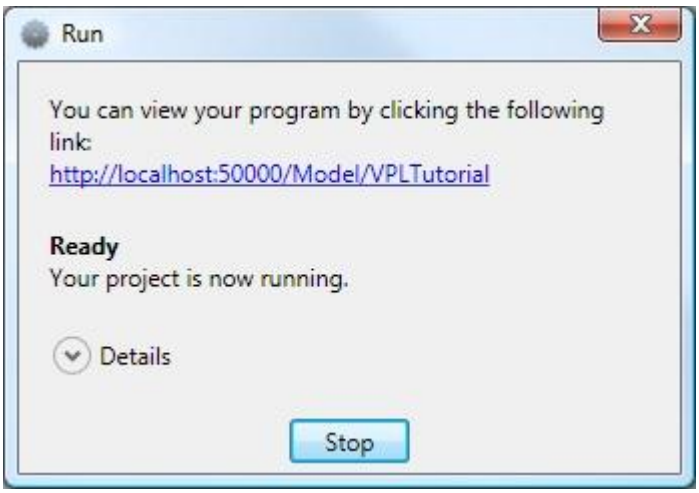
같은 입력 연결(action) 핀에 대해 여러 개의 직접연결을 가질 수 없습니다. 그러나 Merge 또는 Join activity 를 사용해 다중 activity 에서부터 같은 입력 연결 핀으로 메시지를 연결할 수 있습니다.

Activity 는 또한 다중 출력 연결을 가질 수 있지만 이것 중의 한 개만 한번에 연결할 수 있습니다. 만일 Activity 가 결과와 알림 출력 연결 핀을 둘 다 가질 경우, 다른 연결로 이어지는 Activity 의 별도 참조를 사용해야 합니다. 그러나, 당신은 같은 출력 핀에 다중 연결을 만들 수 있습니다.

Join Activity 로 두 개 이상의 알림 연결을 이을 수 없습니다. 이것은 또한 알림 연결을 포함한 둘 이상의 데이터 플로우 연결에서도 그러합니다. 이 경우 Merge Activity 을 사용해야 합니다.

실행과 디버깅

여러분의 VPL 프로젝트를 실행하기 위해, Run 메뉴에서 Start 를 선택하십시오. 이 후 애플리케이션이 시작되며 디버그 추적을 위한 하이퍼링크와 Stop 버튼을 가진 대화 상자가 다음과 같이 나타날 것입니다.



디버그 링크를 클릭하면, 요약 웹 페이지가 표시됩니다. 페이지 위의 버튼을 사용해 다이어그램 조작을 멈추고 단계별로 수행할 수 있습니다.

디버그 페이지에 액세스하기 전에 프로젝트가 종료되는 경우는 위 조작 대신 **Debug Start** 명령을 이용해 최초 블록부터 단계별로 제작된 다이어그램을 실행할 수 있습니다. 디버그 뷰를 이용해 다이어그램 실행 중 메시지 값을 볼 수 있습니다. 디버그 페이지에서 다이어그램을 볼 수 없다면 웹브라우져의 새로고침 명령을 이용해 다시 로드하십시오.

프로그램 종료를 위해 Run 다이얼로그의 **Stop** 버튼을 누릅니다.

여러분의 프로젝트가 하나의 서비스이기 때문에 VPL 은 DSS 런타임을 실행하고 만들어진 서비스를 로드합니다. 이를 위해 HTTP 와 TCP 포트가 필요하며 VPL 은 자동으로 서비스를 위한 디폴트 값을 설정합니다. 이 포트값은 **Run** 메뉴의 Setting 명령으로 바꿀 수 있습니다.

또한 여러분의 애플리케이션에 디버그 정보를 추가하고 이를 웹브라우저에서 확인할 수 있습니다. 이에 대한 링크는 Debug 페이지에 있습니다.

제작된 VPL 애플리케이션은 VPL 개발 환경과 독립적으로 실행가능 합니다.이를 위해 **Build** 메뉴의 **Compile As a Service** 명령을 이용하십시오. 이 명령은 DSS 런타임에서 다음의 방법으로 실행가능한 서비스를 생성합니다:

- 시작 메뉴에서 마이크로소프트 Robotics Studio 아래의 **Run DSS node** 를 실행합니다. 노드가 시작되면 DSS 웹페이지가 열립니다. 여기서 Control Panel 을 클릭해 열고 여러분의 프로젝트 이름을 검색창에 입력해 찾습니다. 찾아진 서비스를 Create 버튼을 이용해 시작합니다.

- 시작메뉴에서 **Microsoft Robotics Studio Command Prompt** 를 실행합니다.아래와 같이 제작된 서비스를 실행할 포트와 서비스의 컨트랙트를 옵션으로 지정하여 Dsshost 를 실행합니다. 서비스의 컨트랙트는 VPL 의 프로젝트 창의 Diagrams entry 에서 확인할 수 있습니다..

```
dsshost.exe                                                    /p:50000
/c:http://schemas.tempuri.org/2006/12/helloworld/diagram.html
```

C# 코드 생성

이전버전의 MVPL 은 간단한 코드 생성기능을 지원했으며, 새 코드 생성 기능은 MVPL 에서 디자인된 다이어그램을 C#코드로 변환해 하나 이상의 DSS 서비스를 가진 어셈블리로 컴파일할 수 있습니다. 이 코드생성 기능을 이용해 C#의 빠른 실행기능을 사용할 수 있습니다.

또한 코드 생성기능은 복잡한 서비스 개발에 있어 MVPL 을 시작으로 사용할 수 있게 합니다.

코드 생성
코드를 생성하는 두계의 단계가 있습니다: 코드 생성 설정과 코드 생성하기

코드 생성 설정

코드생성을 위해 다이어그램은 올바르게 설정되어야 합니다.

1. **Project** 툴박스에서 **Diagrams** 폴더를 클릭합니다.
2. **Properties** 툴박스에서 **KeyLocation** 프로퍼티가 올바르게 strong name key (**.snk** 파일, "samplesWmrisamples.snk")로 설정되어 있는지를 확인합니다.
3. **Properties** 툴박스에서 **SourceLocation** 프로퍼티를 지정합니다. 이 설정은 코드 파일들이 생성될 위치를 지정합니다.

주목: 위 설정은 서비스 래퍼 생성을 위한 C#코드 생성에도 동일하게 적용됩니다.

코드 생성하기

Build 메뉴에서 "**Compile as a Service (No Interpreter)**"를 선택해 "**Create a Service**" 다이얼로그 박스가 표시되게 합니다. 컴파일이 성공적으로 수행되면 "**Compilation complete.**"가 표시됩니다. 만일 서비스 컴파일에 오류가 생기면(알려진 이슈들 참조) 그에 적합한 오류 메시지가 나타납니다.

"**Compile as a Service (No Interpreter)**" 명령이 성공적으로 수행되면 생성되고 bin 디렉터리에 복사됩니다.

SourceLocation 프로퍼티에 지정된 위치에 몇개의 **.cs** 파일과 하나의 **.csproj** 파일이 생성됩니다. 이생성된 코드는 필요할 경우 수정해 사용할 수 있습니다.

주의사항: 코드를 다시 생성하면 경고 없이 바로 기존 위치에 겹쳐서 파일이 생성되므로 주의하십시오.

생성된 서비스 실행

마이크로소프트 Robotics Studio 에 있는 다른 서비스와 같이 생성된 서비스를 실행하는 다양한 방법이 있습니다.

주의사항: 상세한 사항은 알려진 이슈들을 참조하세요.

1. DSS 노드를 시작한 뒤 control panel 에서 생성된 서비스를 실행합니다.
    1. **dsshost /p:50000 /t:50001 실행**
    2. **http://localhost:50000/controlpanel 로 이동**
    3. 서비스 리스트에서 새로 생성된 서비스로 이동
    4. "Create" 버튼을 눌러 서비스 실행
2. 매니페스트를 이용해 서비스를 실행합니다. 새로 생성된 서비스에 대한 컨트랙트를 가지는 매니페스트를 생성합니다. 컨트랙트 지시자(contract identifier)는 생성된 파일 중 **DiagramTypes.cs** 에서 찾을 수 있으며 코드생성 전에 다이어그램 프로퍼티에서 **ContractPrefix** 설정을 고쳐서 바꿀 수 있습니다.
3. MVPL 을 닫고 다시 열어 생성된 서비스를 사용할 새 다이어그램을 만듭니다. 생성된 서비스는 서비스 리스트에 나타납니다. MVPL 에서 해당 서비스를 이용하는 다이어그램을 실행합니다.

알려진 이슈들

- Enums 는 지원되지 않습니다.
- 리스트 함수는 지원되지 않습니다.
- 어떤 상황에서 "merge" 가 뒤에 붙는 "if" 블록은 잘못된 코드 생성을 야기합니다. 만약 이 경우는 코드를 검사해야 합니다.
- 새로 생성된 서비스는 MVPL 의 서비스 리스트에 자동으로 나타나지 않습니다. MVPL 을 닫고 다시 열어야 합니다.
- .NET Compact Framework (CF)을 위한 서비스 코드 생성은 지원되지 않습니다 - 다른 서비스를 .NET CF 에서 실행하도록 바꾸는 방법을 따라 생성된 코드를 직접 바꾸어야 합니다. 생성된 코드는 .NET CF 에서 지원하지 않는 메서드를 사용할 수 있습니다.
- 생성된 서비스에 설정 정보가 포함되지 않습니다. 만약 다른 서비스를 설정하고 오케스트레이션 하는 다이어그램인 경우 초기 상태와 파트너를 직접 설정하거나 매니페스트를 이용합니다:
    1. 다이어그램의 어셈블리를 생성한 후 MVPL 을 닫고 다시 엽니다. 생성된 서비스를 사용하는 새 다이어그램을 만들어 방금 컴파일된 서비스를 드래그 합니다. 서비스를 설정한 뒤 다이어그램을 실행합니다. 새 다이어그램은 코드를 포함하지 않고 컴파일된 서비스를 설정하기 위해 존재합니다.

2. 새로 생성된 서비스를 위한 파트너 설정을 가진 매니페스트를 직접 만듭니다. 만약 이 방법을 이용할 경우 생성된 코드의 파트너 어트리뷰트에서 PartnerCreationPolicy 를 UseExistingOrCreate 에서 UsePartnerListEntry 로 바꾸어야 합니다.

3. 레퍼런스

기본 Activity

마이크로소프트 비주얼 프로그래밍 언어는 데이터 플로우 프로그램을 제작을 도와주는 기본 activity 를 포함합니다. 이 블록은 일반적으로 서비스 블록 사이의 접속에 사용됩니다. 그러나 이들 블록 또한 함께 접속 될 수 있습니다.

Activity

이 Activity 블록은 여러분의 프로그램을 작성하는데 사용되며, 각각은 데이터 플로우 다이어그램의 집합으로 구성되어 있습니다.

Activity는 다른 다이어그램에서 사용시 하나의 블록으로 표시됩니다.

또한 Activity는 서비스 내로 컴파일 되어 다른 서비스에서 사용될 수 있습니다.

Calculate

Calculate Activity 는 단순한 산술 혹은 논리 연산을 수행합니다. 이는 또한 수치, 메시지의 값, 데이터 구성 요소 또는 다른 서비스에 의해 제공된 미리 결정된 값들이 입력되어 표현됩니다.

사용할 수 있는 수치 데이터:

+	Add
-	subtract/minus
*	Multiply
/	Divide
%	Mod

플러스(+) 연산자는 또한 문자열을 연결하기 위해 사용됩니다. 이는 또한 텍스트(문자열)와 수치 자료를 이중 인용부(예 . “The answer is ” + x/4)를 사용하여 조합할 수 있습니다.
사용가능한 논리 연산자:

&&	AND
||	OR
!	NOT

괄호를 이용해 입력하는 표현식의 우선 순위(평가의 순번)을 지원합니다.

Calculate 박스의 텍스트 박스를 클릭하여 입력 메시지, 데이터 구성 요소 및 사전 정의의 값들을 포함한 리스트를 표시합니다.

Comment

Comment Activity는 텍스트 블록을 다이어그램에 추가하는 기능입니다.

텍스트 박스에서 표시하기 원하는 텍스트를 입력하십시오.

Comment 블록은 연결을 지원하지 않으며 다이어그램에 어느 곳이라도 이를 배치할 수 있습니다.

Data

Data activity 는 간단한 데이터 값을 다른 activity 나 서비스에 보낼 때 사용합니다. 데이터의 형식을 지정하기 위해 텍스트 박스 아래의 드롭 다운에서 타입을 선택하고 텍스트박스에 값을 입력합니다.

If

If activity 는 입력된 메시지에 대해 주어진 조건에 따라 출력을 선택하게 합니다. 만약 조건이 true 이면 입력 메시지와 그 데이터를 첫째 출력 연결로 보냅니다. True 가 아닐 경우 Else 출력이 사용됩니다.

조건 표현은 다음 연산자를 사용합니다:

= or ==	Equals
!= or <>	not equals
<	less than
>	greater than
<=	less than or equals
>=	great than or equals

Calculate activity 의 연산자 또한 이용가능 합니다. 이 때 전체 문장이 ture 혹은 false 로 계산됩니다.

Activity 블록에서 Add(+)버튼을 클릭해 activity에 조건을 추가합니다.

Join

Join activity 는 두 데이터 플로우를 연결합니다.

Activity 가 데이터를 보내기 전에 입력 연결에서 받은 메시지의 데이터가 결합되고 모든 입력 연결에서 메시지들이 받아져야 된다는 점에서 Merge 와 다릅니다.

텍스트 박스에 입력한 텍스트가 메시지를 표현하는 로컬 변수 이름입니다. 이 변수를 데이터 멤버를 참조하는데 직접 도트 표기(예 x.a)를 이용할 수 있습니다.

List

List activity는 데이터 아이템을 위한 빈 list를 생성합니다.

List 를 만들려면 activity 블록 혹은 프로퍼티 뷰의 드롭 다운 리스트에서 아이템을 위한 타입을 선택합니다.

List에 항목을 추가하기 위해 List Functions activity를 사용합니다.

여러분의 다이어그램 이외에서 list 를 사용하기 위해 저장하려면 Variable activity 를 이용해 list 변수를 생성합니다.

List Functions

List Functions activity는 기존 list 편집을 가능하게 합니다.

필요한 list에 적용할 함수를 선택하기 위해 블록에서 드롭 다운 리스트를 사용합니다.

Merge

Merge activity 는 단순히 둘 이상의 데이터를 함께 합쳐줍니다. Merge 에서는 메시지가 통과될 때 어떤 조건이나 의존성이 없습니다. Activity 의 역할은 단순히 메시지를 다음 activity 로 전달하는 것입니다.

Switch

Switch activity 는 입력된 메시지와 텍스트 박스에 입력된 표현과의 호환성을 기반으로 이를 연결합니다.

Activity block에 Add(+)버튼을 클릭해 Case 분기(일치 조건)를 추가할 수 있습니다.

Variable

Variable activity 는 변수를 생성하고 그 값을 쓰고 읽게 해줍니다. 텍스트 박스 주변에서 드롭 버튼을 클릭해 나타나는 드롭 다운리스트에서 variable을 선택할 수 있습니다.

어던 변수도 정의하지 않았고 새 변수를 만들기 원하면 리스트(혹은 Edit 메뉴에서)에서 Define Variables 를 선택합니다. 이것은 여러분이 변수를 추가하고 그 타입을 정의할 수 있는 Define Variables dialog box를 보여줍니다. 변수 타입에는 리스트의 타입도 포함됩니다.

변수 이름은 대소문자 구분입니다. 변수를 고칠 때 주의하십시오. 이름은 문자로 시작하고 알파벳과 숫자로만 조합됩니다.

Variable activity는 값을 읽기위한 GetValue연결과 값을 쓰기위한 SetValue 연결을 지원합니다.

데이터 타입

마이크로소프트 비주얼 프로그래밍 언어는 .Net Visual C# 스타일의 데이터 타입을 지원합니다.

VPL Type	Description
bool	Boolean values: true, false
byte	8 bit unsigned integer
sbyte	8 bit signed integer
char	character
decimal	fixed point decimal number
double	double precision floating point number
float	single precision floating point number
int	32 bit signed integer
uint	32 bit unsigned integer
long	64 bit signed integer
ulong	64 bit unsigned integer
short	16 bit signed integer
ushort	16 bit unsigned integertd>
string	character string (text)

VPL 메뉴

File

New - 새 프로젝트를 만듭니다.

Open - 기존 프로젝트 파일을 엽니다.

Save - 현재 프로젝트를 저장합니다.

Save As - 프로젝트를 지정된 이름으로 저장합니다.

Recent Projects - 최근 사용된 프로젝트 파일을 보여주고 열수 있습니다.

Exit -VPL 을 종료합니다.

Edit

Undo - 마지막 편집 되돌림

Redo - 마지만 되돌림 취소

Cut - 선택된 객체 잘라냄 (클립보드에 저장).

Copy - 선택된 객체 복사 (클립보드에 저장).

Paste - 마지막 cut 혹은 copy 된 객체를 추가

Del - 선택된 객체 제거

Actions and Notifications - 사용자 activity 의 동작, 결과, 알림을 추가 및 변경하게 해주는 다이얼로그 박스 표시

Define Variables - 변수와 그 타입을 정의하는 다이얼로그 박스 표시

Connections - 두 activity 사이의 연결을 편집하는 다이얼로그 박스 표시

Data Connections - 데이터 연결을 편집하는 다이얼로그 박스 표시

Set Configuration - 서비스 설정과 파트너 설정을 정의하는 configuration pane 표시

View

Toolboxes - 기본 Activity, 서비스, 프로젝트와 프로퍼티 창 표시/숨김

Grid - 다이어그램을 위한 그리드 표시/숨김

Toolbar -VPL 툴바 표시/숨김

Build

Compile as a Service - 현재 프로젝트의 다이어그램과 유저 activity를 서비스로 컴파일 함

Run

Start - 현재 프로젝트 실행

Debug Start - 현재 프로젝트를 시작하고 첫째 activity 에서 멈추고 디버그 뷰를 표시

Port Settings - 실행할 때 사용할 포트를 설정하는 다이얼로그 표시

Help

Contents -VPL 헬프 파일 표시

About -VPL의 저작권과 버전 정보 표시

4. VPL 튜토리얼

VPL 튜토리얼 1 - Hellp World

이 튜토리얼은 "Hello World"를 표시하는 과정을 소개합니다. 마이크로소프트 비주얼 프로그래밍 언어에서 지원하는 Data 블록과 Simple 다이얼로그 블록을 사용하여 문자열을 출력합니다.

Hello Word 다이어그램을 생성하는 방법

VPL 을 실행하기 위해서는 **시작**메뉴에서 **마이크로소프트 Robotics Studio** 폴더에 있는 **Visual Programming Language**를 클릭 하십시오.

File 메뉴에서 **New** 을 클릭하셔서 새로운 프로젝트를 만드십시오. 이제는 Basic Activities toolbox 에 있는 Data Activity 를 더블 클릭하시거나 마우스로 끌어다가 입력하십시오. 드롭다운 리스트에서 **string** 을 선택하십시오. Data 블록에 있는 텍스트 상자를 클릭하신 후 Hello World라고 입력하십시오.

그림 1 - Data 블록

다음으로 Simple 다이얼로그 Activity 를 Services toolbox 에서 마우스로 끌어다가 Data Activity 블록 옆으로 가져다 놓으십시오. (어떤 서비스를 찾는 대 걸리는 시간을 단축시키기 위해서 toolbox 상단에 원하는 블록을 검색하셔서 찾을 수도 있습니다.)

다음으로 Data 블록의 출력 연결 핀으로부터 Simple 다이얼로그 블록까지 마우스로 끌어서 연결하십시오. 이 작업 후에 Connections 다이얼로그 창이 표시됩니다. **DataValue** 를 첫번째 항목에서 고르고 **Alert 다이얼로그**를 두 번째 항목에서 선택한 후 **Ok 버튼을** 클릭합니다.

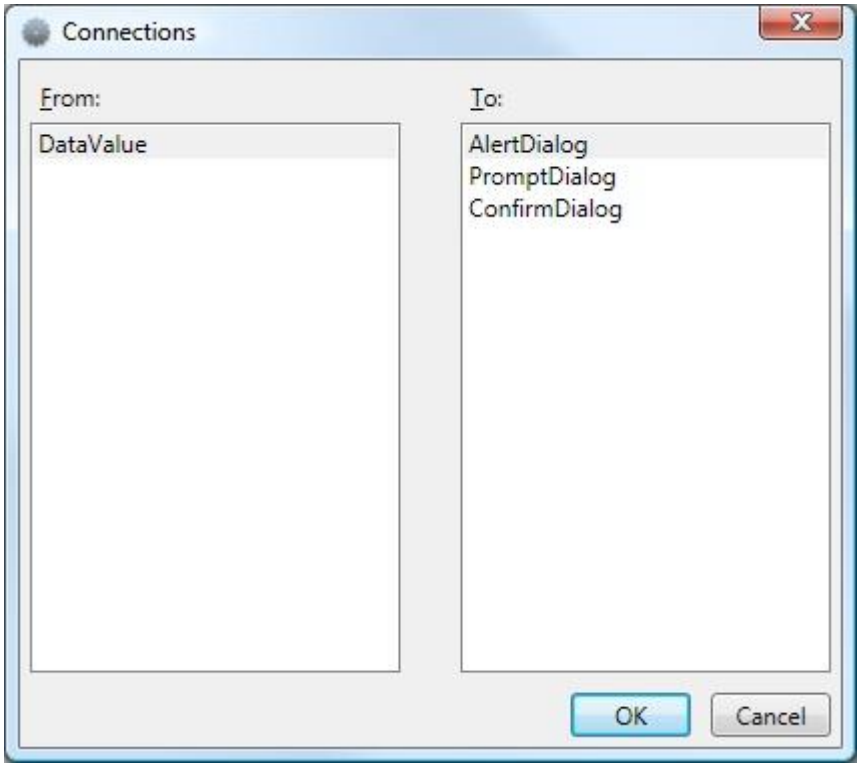


그림 2 - Connection 다이얼로그 박스

Data Connection 다이얼로그 창이 실행된 후 드롭다운 리스트에서 **Value** 를 선택하세요.

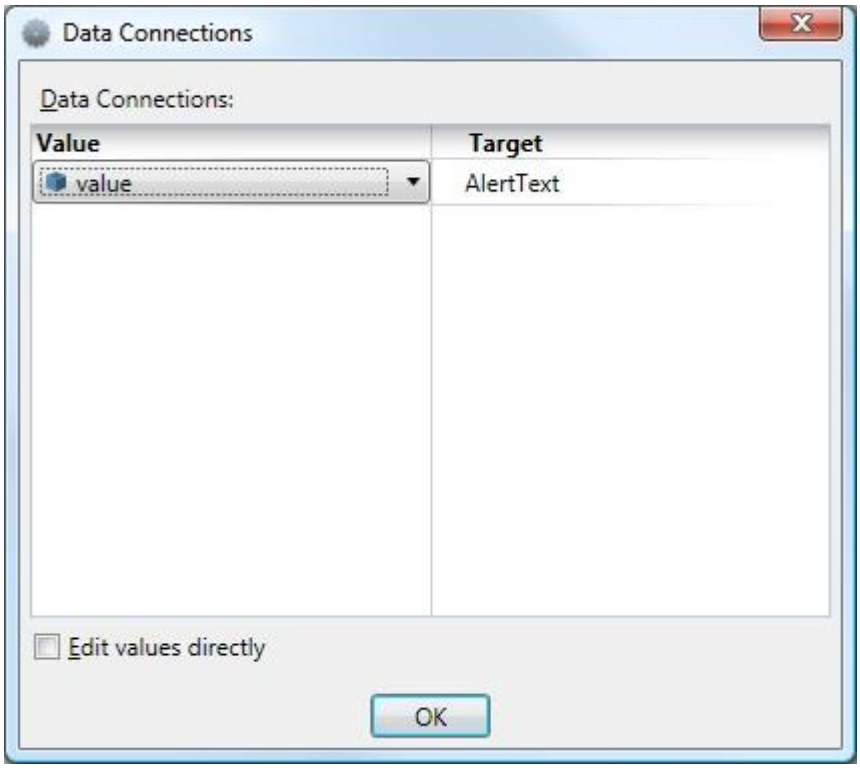


그림 3 - Data Connections 다이얼로그 창

이렇게 함으로 인해서 VPL 에게 Data Activity 에 있는 값을 Alert Form 의 메시지 텍스트로 보내겠다고 입력을 하게 된 것입니다.

다이어그램은 아래와 같이 나타날 것입니다.

그림 4 - VPL 다이어그램

그런 다음 **Run** 메뉴에 있는 **Run** 명령어를 누르시면 됩니다. (또는 **F5** 를 누르셔도 됩니다.) 만약 아직 프로젝트를 저장하지 않았다면 VPL 이 저장 다이얼로그를 엽니다. 프로젝트 이름을 입력하신 후 **Save** 를 누르시면 됩니다.

이제 VPL 이 애플리케이션을 실행시킵니다. 만약 애플리케이션을 unblock 하겠냐고 묻는 메시지가 뜬다면 **Unblock** 을 누르십시오.

모든 과정을 마치셨다면 **Hello World** 가 쓰여져 있는 simple alert 다이얼로그 상자가 나타날 것입니다.

그림 5 - Completed Alert 박스

애플리케이션을 멈추고 싶으시다면 Run 다이얼로그에 있는 **Stop** 버튼을 누르시면 됩니다.

VPL 으로서의 첫 작품을 완성 시켰습니다. 계속해서 VPL 튜토리얼 - 값을 증가시키기에서 VPL 과 데이터 흐름 제어 activity 에 대해 더 자세히 알아보도록 하겠습니다.

VPL 튜토리얼 2 - 값 증가시키기

이 튜토리얼에서는 마이크로소프트 비주얼 프로그래밍 언어가 지원하는 몇 가지 다른 기본적인 데이터 플로우 제어 activity 들에 대해 설명합니다. 어느 변수를 생성, 초기화 시키고 그 다음 Text-to-Speech 를 사용하여 값을 스피커로 출력시키며 열까지 반복합니다.

다이어그램을 만들어 계산하는 방법

우선 **File** 메뉴에서 **New** 를 선택한 후 새로운 프로젝트를 만듭니다. 그런 후 Toolbox 에서 Variable activity 를 마우스로 끌어다 놓거나 더블클릭 해서 입력합니다.

위의 작업 후에는 **Define Variable** 다이얼로그를 열기 위해 Variable activity 블록에 있는 ellipsis 버튼은 누르십시오. 창이 열리면 그 곳에 있는 **Add** 버튼을 누른 후 텍스트 상자에 Test 라고 입력합니다. **Type** 드롭다운 리스트에서는 꼭 *int* 로 설정되어있는지 확인하셔야 합니다. 확인이 끝나셨다면 **Ok** 를 클릭합니다.

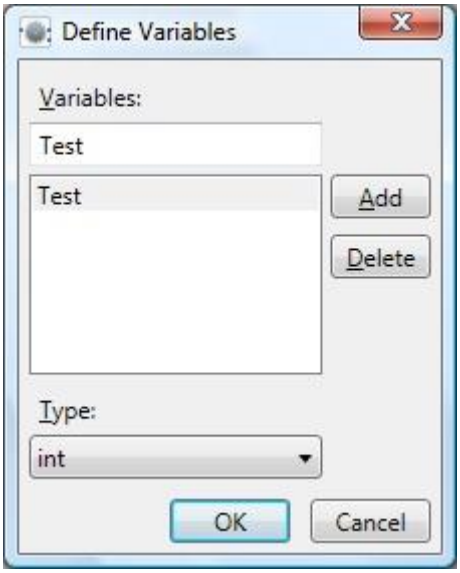


그림 1 - Variables 다이얼로그 박스

만약 Variable 텍스트박스에 현재 설정으로 Test variable 이 나타나지 않는다면, 다시 한번 드롭다운 리스트를 열어서 새롭게 만든 variable 으로 설정하십시오.

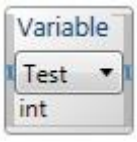
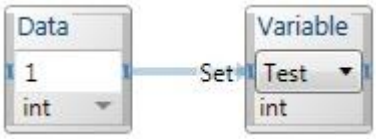


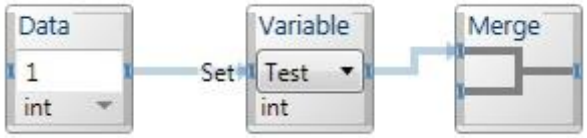
그림 2 - Variable 텍스트 박스

다이어그램에 새로운 Data 블록을 왼쪽에 추가해서 그것을 Variable activity 블록과 연결하십시오. Connections 다이얼로그 상자가 열릴 것입니다. Output 연결 핀은 **Data** 로 설정하시고 Input 연결 핀은 **Set Value** 로 설정하신 다음에 **Ok** 를 누르면 됩니다.

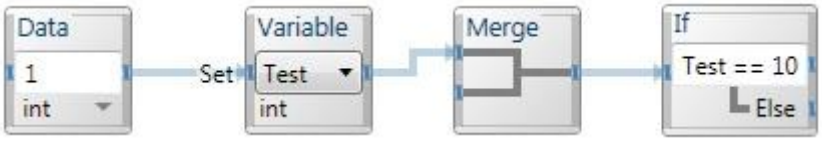


드론다운 리스트에서 int 를 선택하고 Data 블록에 있는 텍스트 상자에는 1 을 입력하십시오. 이렇게 하면 data 와 type 을 둘 다 설정하게 되는 것입니다. 연결은 그 다음에 Test 를 1 로 초기화 할 것입니다. Set Value 으로 Variable 블록을 연결하게 되면 그것은 스스로 값을 1 로 설정할 뿐만 아니라 그 variable 을 output 연결로 보내기도 합니다.

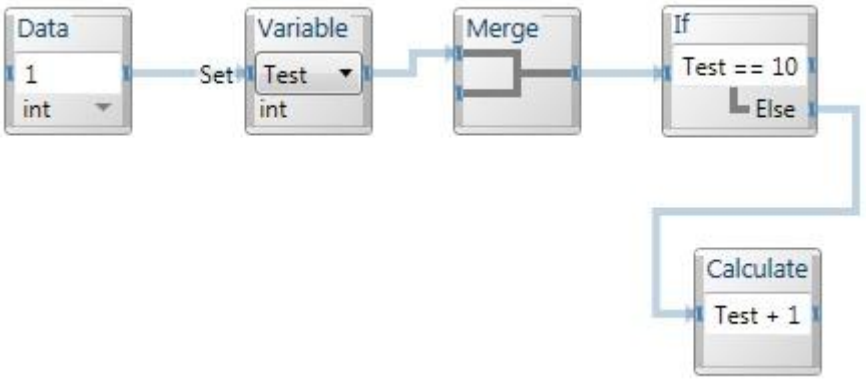
Merge 블록을 Variable 블록과 연결 하십시오. 이 블록을 사용해서 counting loop 를 만들 것입니다. Merge 블록은 여러 개의 input 을 가질 수 있습니다.



그 다음 If activity 를 Merge 블록의 오른쪽에 추가하십시오. Merge 블록의 오른쪽 핀과 If 블록의 왼쪽 핀과 연결하십시오. If activity 블록에는 Test==10 을 입력 하십시오. 이렇게 함으로 인해서 If activity 가 하는 일은 variable 이 10 이 되는 지를 알아보는 것입니다.

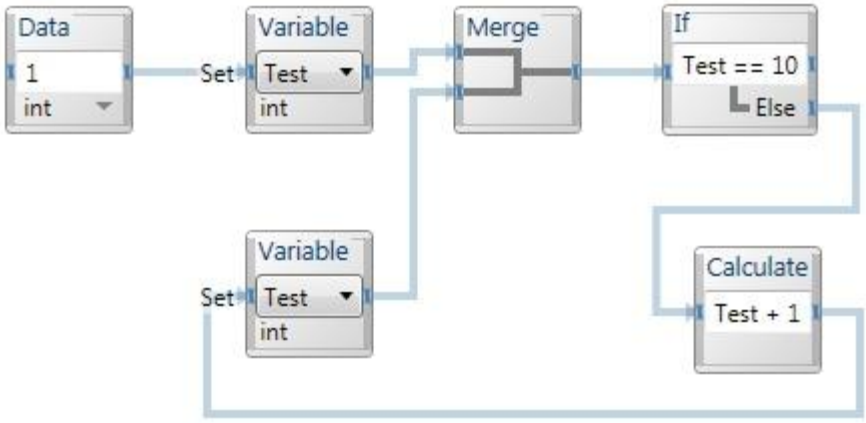


다음으로 Calculate 블록을 추가하셔서 If activity 블록의 Else 부분과 연결하십시오. Else 의 의미는 만약 If activity 의 조건에 만족하지 못할 경우 Calculate 블록을 실행하겠다는 의미입니다. Calculate 블록에는 Test + 1 을 입력하십시오.

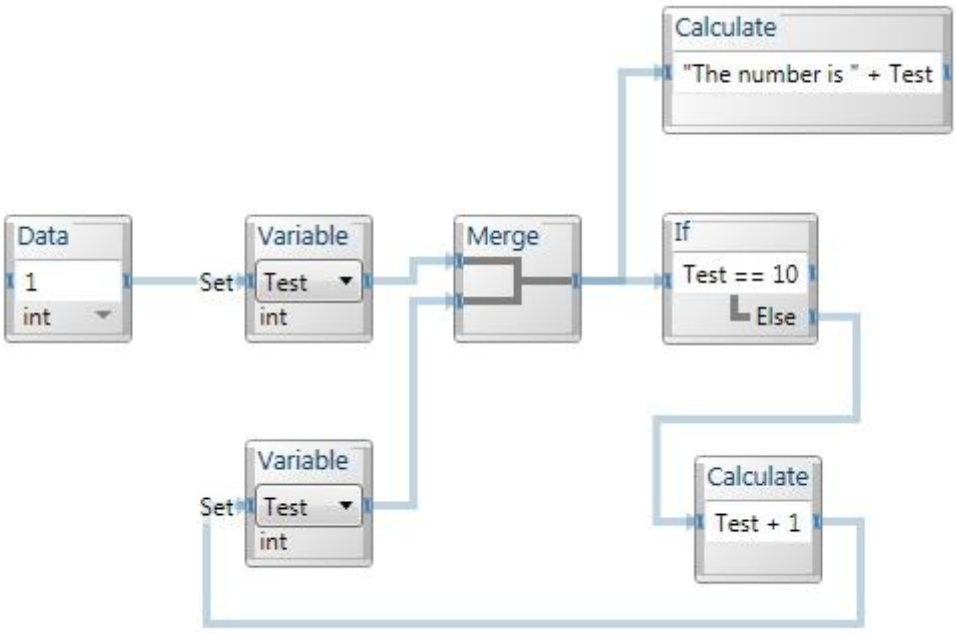


이것은 Test 값에 1 을 더하는 역할을 합니다. 이 더해진 값을 다시 Test 의 값으로 업데이트 하기 위해서 Variable 블록을 하나 더 연결 합니다. (드롭다운 리스트에서 Test 를 선택하는 것을 잊으면 안됩니다.) 그런 후 Calculate 블록과 연결하시면 Connection 다이얼로그가 표시될

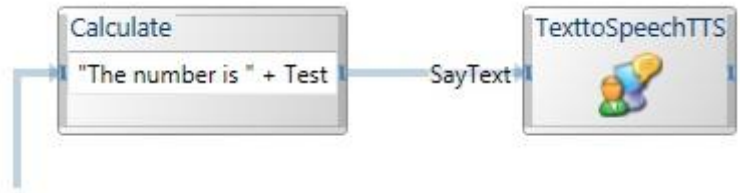
것입니다. **Result Output** 과 **Set Value** 를 선택한 후 **Ok** 를 누르면 됩니다. 이제는 loop 을 완성하기 위해 두 번째 Variable 블록과 Merge 블록의 남은 핀을 연결 시켜 주면 됩니다.



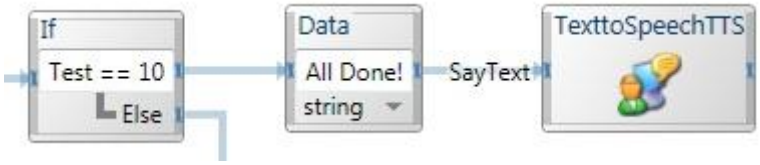
한 단계 더 나아가서 또 하나의 Calculate 블록을 Merge 블록의 오른쪽 핀에 연결하겠습니다. 텍스트 상자 안에 *"The number is" + Test.* 를 입력하면 자동적으로 *Test* 의 값을 텍스트로 변환시켜서 다른 글자들과 합쳐집니다.



Calculate 블록의 오른쪽 단자에 **Text to Speech** 블록을 연결하십시오. 이 둘의 연결 관계를 *CalculateResult* 로 부터 *SayText* 로 바꾸고 Data connection 는 **Value** 로부터 *SpeechText* 로 바꾸십시오. 이렇게 함으로 인해서 text-to-speech engine 이 매 루프 마다 실행되게 됩니다.



마무리하기 위하여 Data activity 블록을 If 블록의 마지막 단자에 연결하십시오. 드롭다운 리스트에서는 **string** 을 고르신 후 *All done!*을 텍스트 상자에 입력하십시오. TextToSpeech 블록을 하나 더 추가 하신 후 연결 관계는 전번처럼 **DataValue** 에서 **SayText** 로 설정하시고 Value 도 SpeechText 로 설정하십시오.



완성된 다이어그램은 위와 같이 보일 것입니다.

모든 것이 잘 연결되어 있다면 실행할 수 있습니다. (**Run** 메뉴에서 **Run** 을 클릭 또는 **F5** 를 누르면 실행됩니다.) PC 가 1 부터 10 까지 세고 다 센 후 **All done!**이라고 말하는 것을 들을 수 있습니다. 만약 올바르게 실행되지 않는 다면 연결들을 다시 확인하시거나 스피커의 볼륨을 조절해 보십시오. (스피커가 아예 없으신 분은 튜토리얼 1 에 나온 Simple 다이얼로그 service 를 Text To Speech 대신 사용하시면 됩니다.)

VPL 튜토리얼 3 - 자신만의 Activity 만들기

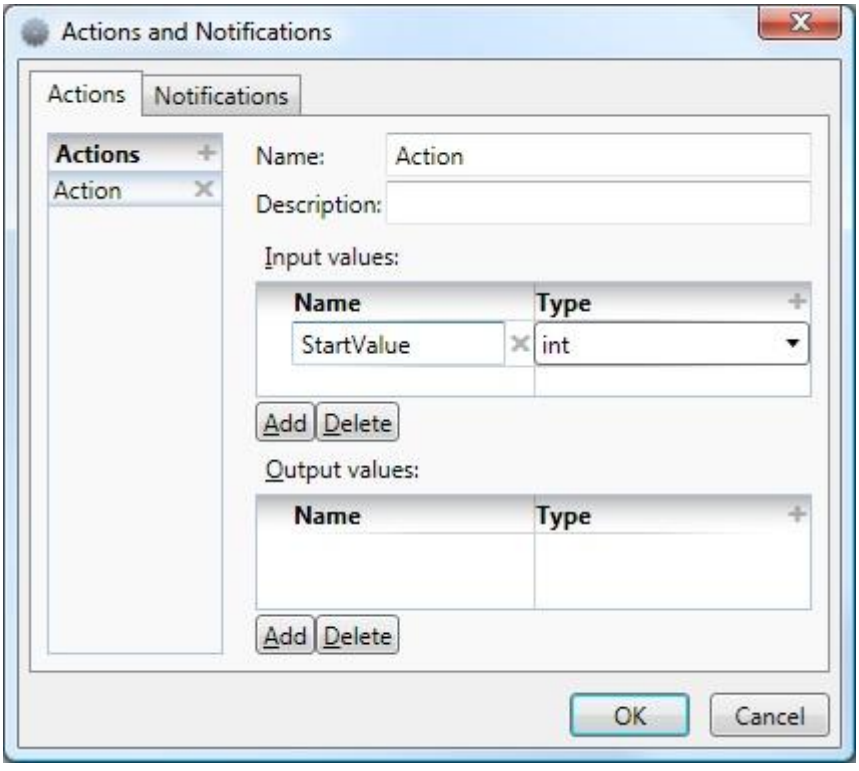
이 튜토리얼은 두번째 튜토리얼과 같은 시나리오를 사용하지만 자기 자신만의 애플리케이션을 모듈화 하는 과정을 소개합니다.

Activity 를 만드는 방법

새로운 프로젝트를 생성한 후 (**File** > **New**) Data activity 를 추가 하십시오. Data activity 의 값을 1 로 설정하십시오.

이제 새로운 Activity 를 만들기 위해 다이어그램에 Activity 블록을 추가 하십시오. 제목을 클릭한 후 Count to 10 이라고 바꾸고 더블 클릭을 하거나 팝업 창에 있는 **Open** 메뉴를 사용하여 해당 별도의 탭 화면으로 오픈합니다. Action 드롭다운을 사용해서 **Action** 엔트리를 선택하십시오. Action 헨들러 페이지가 뜰 것입니다. 다이어그램 페이지는 위의 것과 비슷한 모양이지만 테두리에 새로운 연결 단자들이 있을 것입니다. 그것들은 이 Dataflow 와 다른 Dataflow 를 연결시켜주는 역할을 합니다.

Edit 메뉴에 있는 **Actions and Notifications** 명령어 또는 action selector 옆에 있는 버튼을 누르면 창에 *Action* 탭이 있습니다. 그 탭을 누르고 input value 를 입력하면 이것이 나중에 계산할 때 시작하게 되는 값이 되는 것입니다. Input value 의 이름을 *StartValue* 로 수정하고 값의 타입을 *int* 로 수정한 후 **Ok** 를 누르십시오.

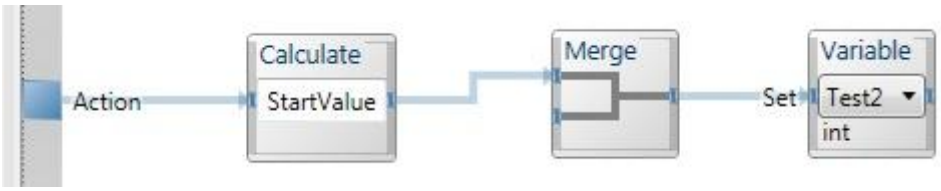


Activity 의 내부에 사용되게 될 Dataflow 는 튜토리얼 2 에서 만든 것을 사용하면 됩니다. Merge, If, Calculate, Variable 와 Data 사용할 예정이지만 약간은 다르게 배열 할 것입니다.

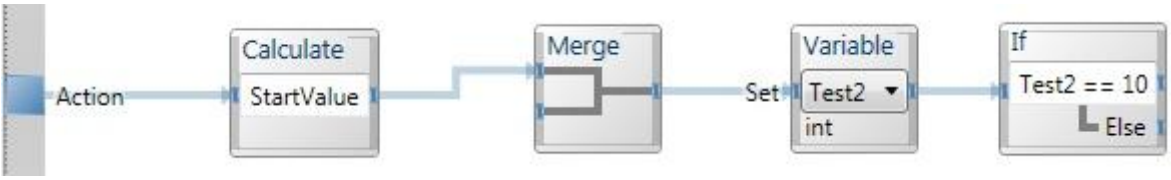
왼쪽 가장자리의 단자 근처에 Calculate 블록을 추가해서 연결시킨 후 텍스트 상자에 *StartValue* 라고 입력하십시오. Actions input message 에서 StartValue 를 꺼내는 작업입니다.

Calculate 블록 옆에 Merge 블록을 연결 시킵니다. 다음으로 Variable 블록을 Merge 옆에 추가합니다. Merge 를 통해 들어오는 Variable 로 Activity 에서 사용할 Variable 을 설정하십시오.

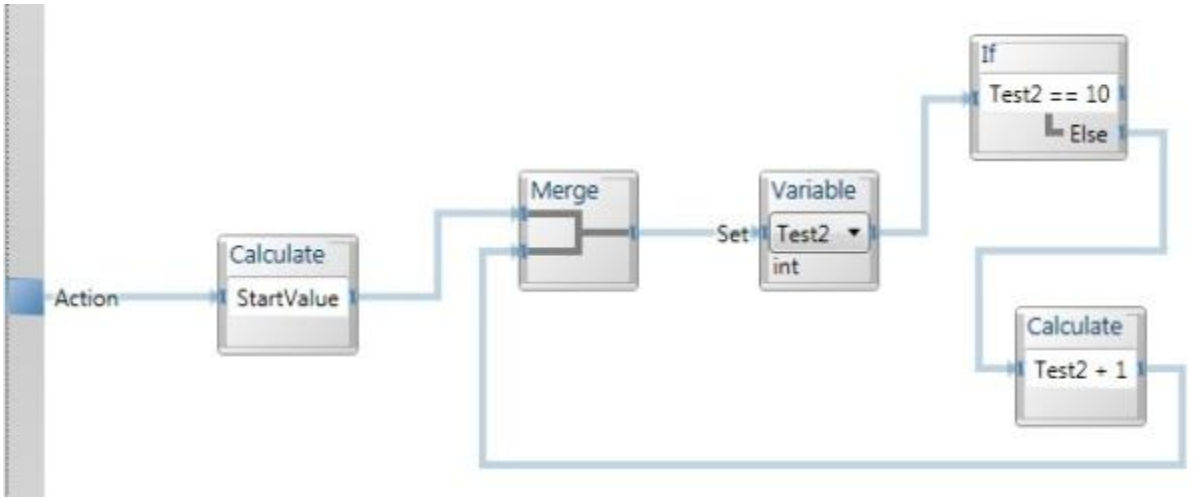
Test2 라는 새로운 Variable 을 생성하십시오. Variable 을 생성하기 위해서는 Variable 블록에 있는 드롭다운 리스트를 통해 **Define variables** 를 선택합니다. 다이얼로그 창이 표시되면, 이름과 Variable 의 타입을 *int* 로 선택합니다. Merge 블록과 Variable 블록을 연결하십시오. Merge 블록을 통해 들어오는 값은 *Test2* 로 설정됩니다.



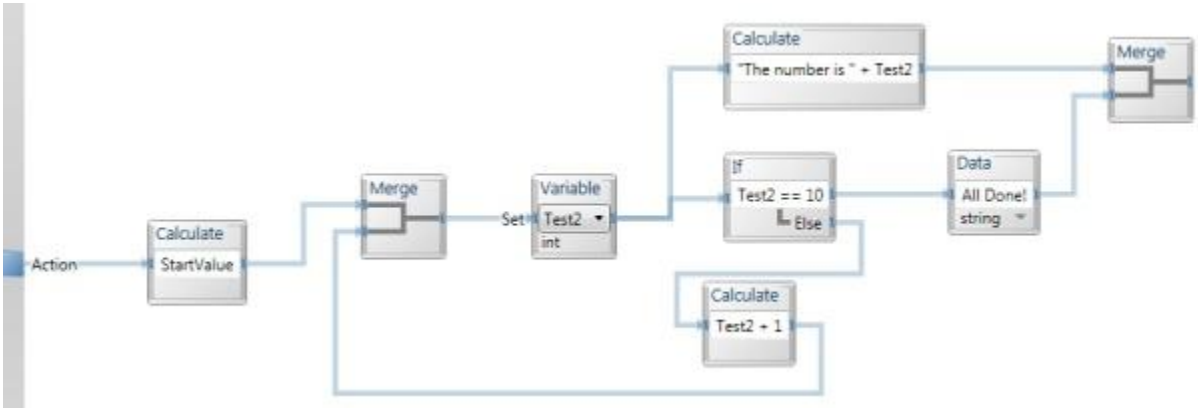
Variable 블록을 If 블록과 연결시킵니다. If 블록의 텍스트 상자에는 *Test2 == 10* 라고 입력하십시오.



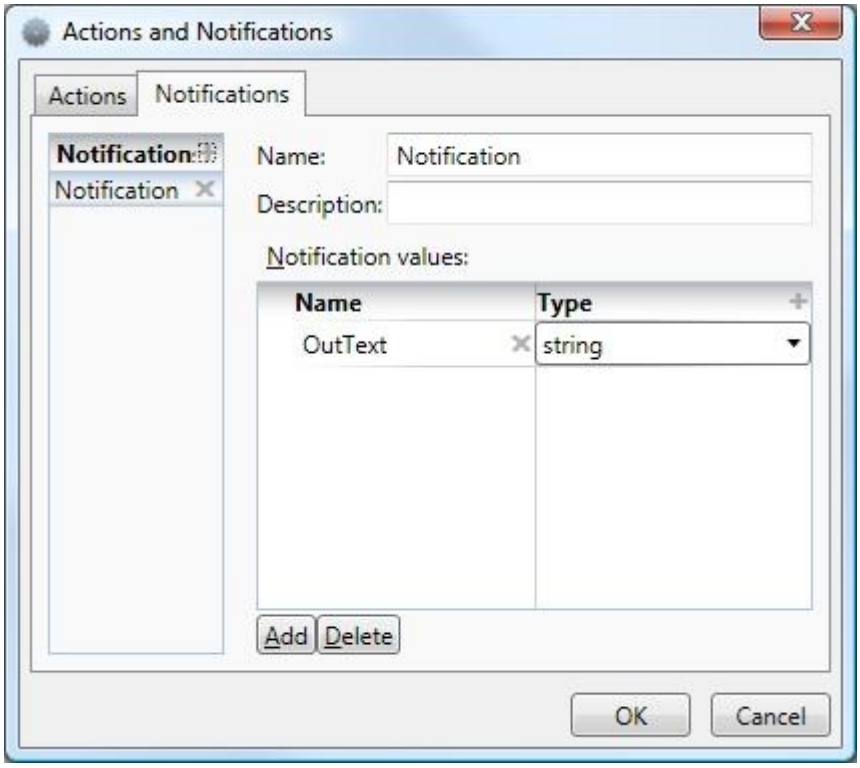
Calculate 블록을 생성하여서 If 블록의 Else 단자에 연결 시킵니다. Calculate 블록에는 *Test2 + 1* 라고 입력합니다. *Test2* variable 을 업데이트 시키기 위해서 Calculate 블록을 Merge 블록과 연결 시킵니다.



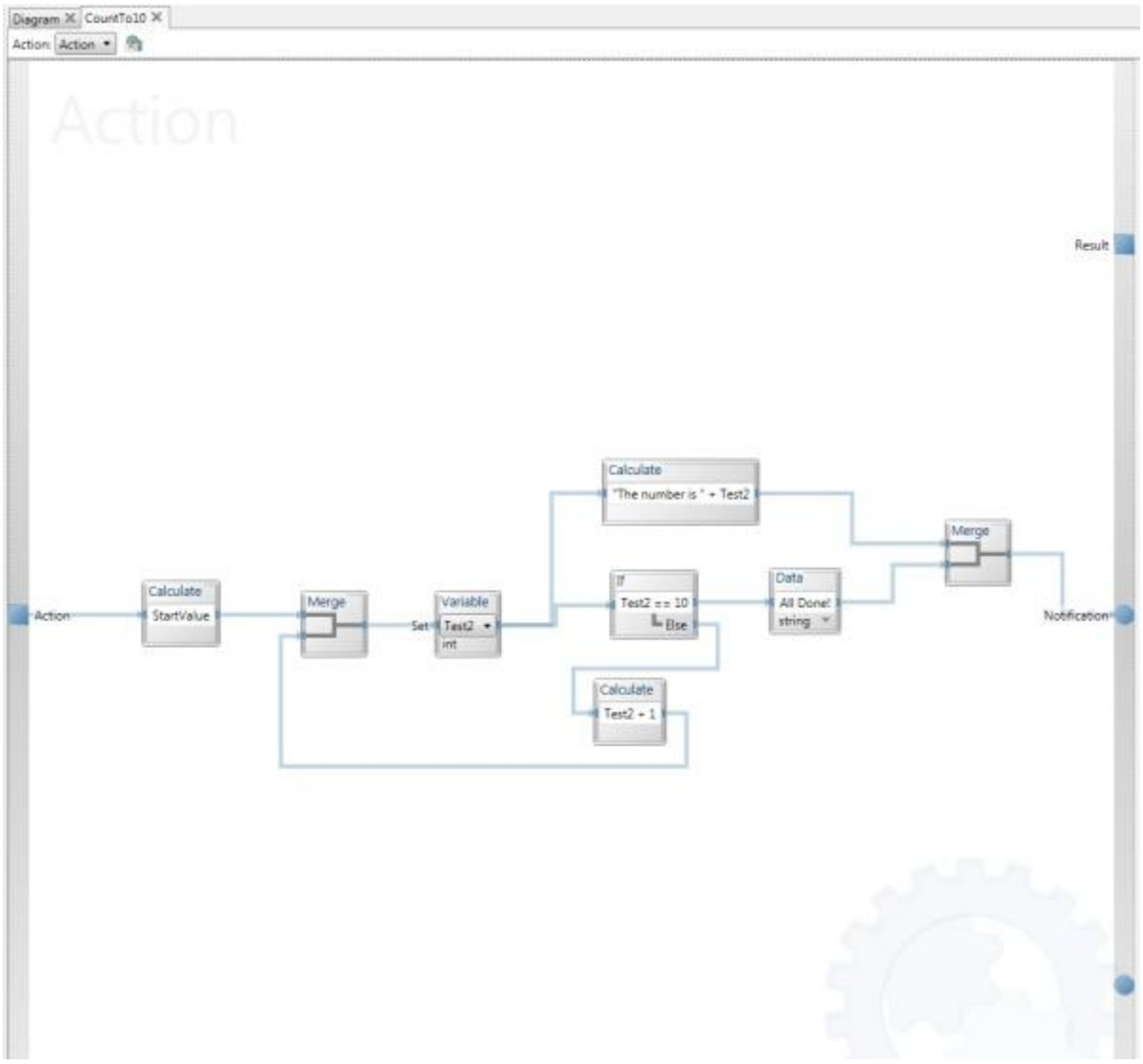
튜토리얼 2 에서 했던 것 과 동일하게 Variable 블록에 Calculate 블록을 연결시킨 후 *"The Number is" + 1* 을 입력합니다. If 블록의 위쪽 단자에는 Data 블록을 연결시켜서 드롭다운 리스트에서 **Text** 를 선택한 후 *All done!*을 입력해 주십시오. 또 하나의 Merge 블록의 두 input 단자에 Calculate 블록과 Data 블록을 각각 하나씩 연결시킵니다.



Dataflow 의 반대쪽 단자를 연결하기 위해서 Edit 메뉴의 **Actions and Notifications** 를 선택합니다. 이번에는 **Notifications** 탭을 누른 후 Notification 옆에 있는 plus sign (+)을 클릭하십시오. **Add** 를 눌러 새로운 *OutText* 를 추가해서 타입을 *string* 으로 설정한 뒤 **Ok** 를 클릭하십시오.



이제는 Merge 블록의 오른쪽 단자를 외각에 있는 Notification 단자와 연결할 수 있습니다. 방금 만든 Notification 을 선택하십시오. Activity 의 dataflow 는 아래와 같이 보여져야 합니다.



Activity 페이지를 닫거나 Dataflow 탭을 눌러서 메인 dataflow 페이지로 돌아 옵니다. Data activity 를 새로운 CountTo10 Activity 와 연결시킵니다.

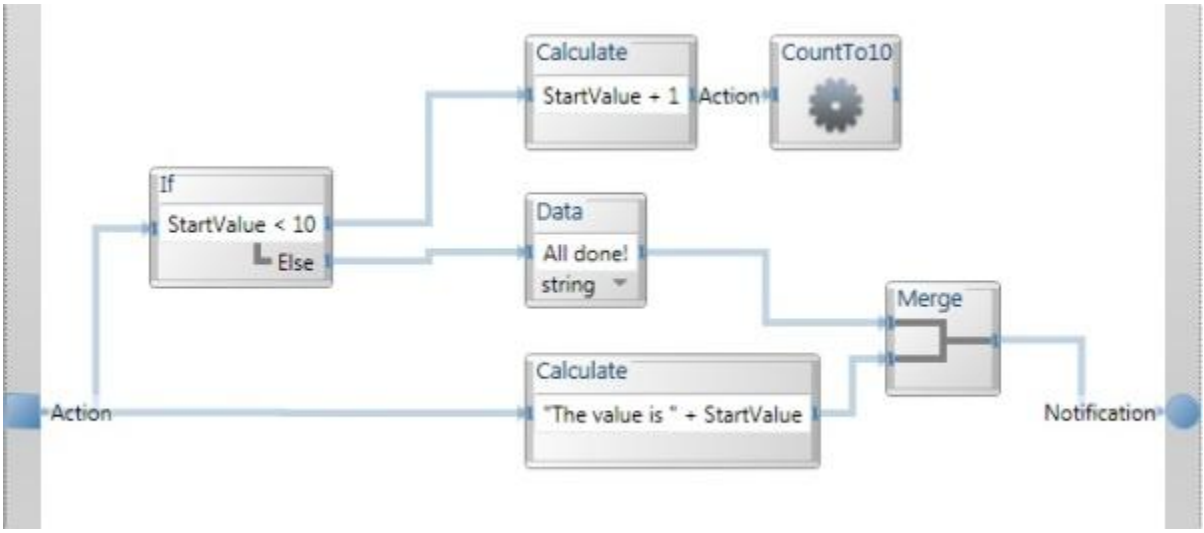
복사와 붙여넣기를 사용해서 Activity 를 하나 더 추가 하십시오.

마지막으로, CountTo10 activity 의 Notification 단자와 TextToSpeech 블록을 추가해서 연결 시킵니다. 연결 관계를 Notification 에서 SayText 로 그리고 Data Connection 을 *OutText* 에서 *SpeechText* 로 바꾸십시오.

Run 메뉴 또는 F5 키를 사용해서 실행 시킵니다. 이 애플리케이션은 튜토리얼 2 와 같은 결과를 가지게 될 것입니다.

Loop 나 Variable 없이 횟수를 계산하는 방법

이 튜토리얼은 Loop 와 Variable 을 어떻게 생성하고 어떻게 사용하는 것인지를 설명합니다. 하지만 많은 경우에는 이런 것들이 사실 별로 필요하지 않습니다. Loop 는 몇 가지 경우에 필요합니다. 다음의 방법은 Loop 나 Variable 을 사용하지 않고 훨씬 더 간단한 방법으로 횟수를 계산할 수 있습니다. 이 예제의 세부적인 과정은 독자들의 연습을 위해 설명하지 않겠습니다.



VPL 튜토리얼 4 - VPL 에서 시뮬레이션 실행

이 튜토리얼은 VPL 을 사용하여 시뮬레이션을 실행하는 방법을 소개합니다. 예제로써 어떤 로봇을 시뮬레이션 상에서 제어하는 프로젝트를 VPL 로 만들 것입니다.

다이어그램을 통한 시뮬레이션 실행

File 메뉴에서 **New** 를 클릭한 후 새로운 프로젝트를 시작하십시오. 다음으로 Services Toolbox 로부터 **XInput controller** 를 더블 클릭 또는 마우스로 끌어서 넣습니다. 만약 Xbox Controller 가 없다면 로보틱스 튜토리얼 4 (VPL) - Drive-By-Wire 에 나오는 대로 "Drive-By-Wire" 인터페이스를 사용하시면 됩니다.

Generic differential Drive Service 를 추가해서 Notification 의 Output 단자와 XInput Controller 를 연결 시키십시오. Connection 다이얼로그 창이 뜨면 **TriggersChange** 에서 **SetDrivePower** 로 설정을 바꿉니다. Data Connection 에서는 **LeftWheelPower** 는 **Right** 로, **RightWheelPower** 는 **Left** 로 설정하십시오.

아래의 화면을 참고하기 바랍니다.

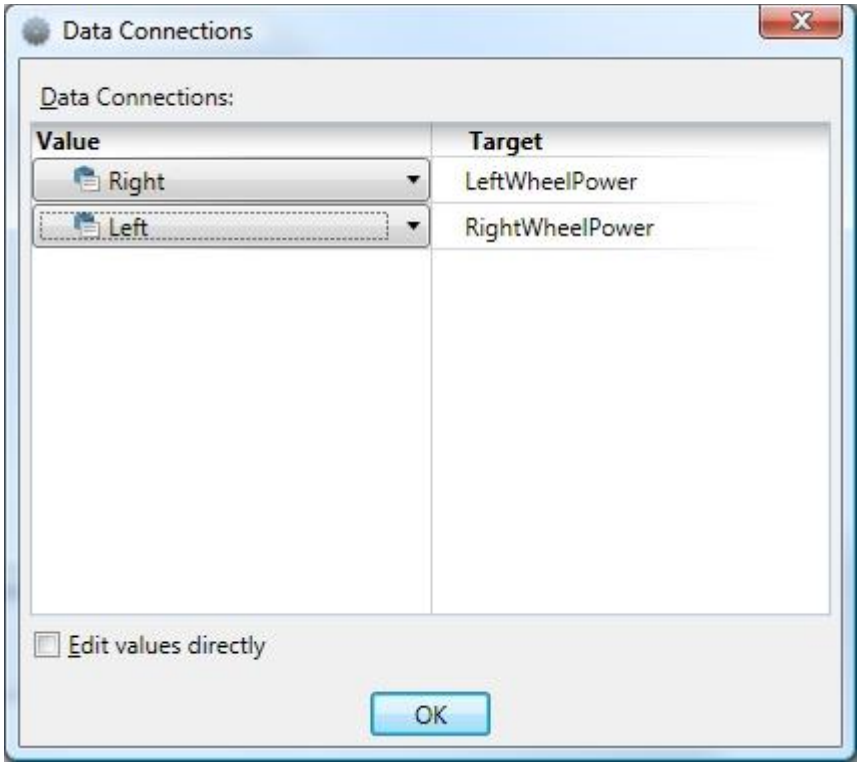


그림 1 - Data Connections 다이얼로그 창

그림 2 - VPL 다이어그램

Direction 다이얼로그를 사용했다면 다이어그램은 아래와 같이 표시될 것입니다.

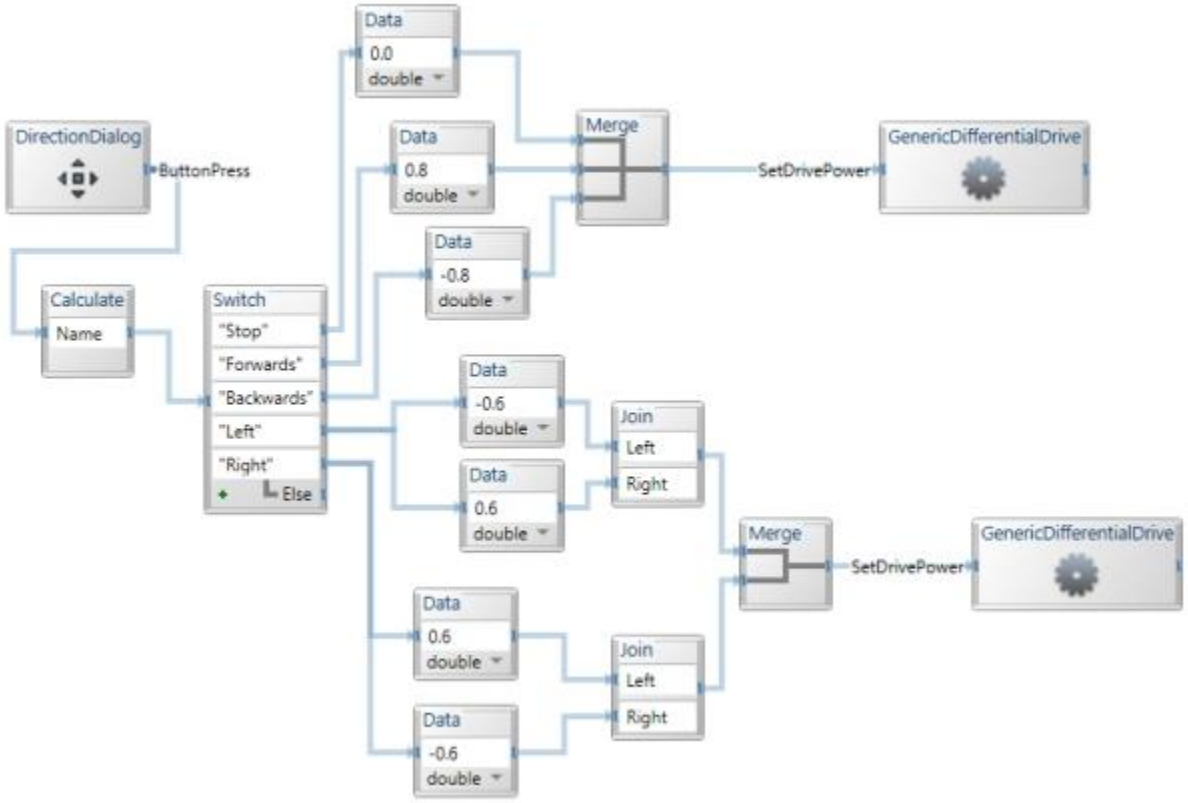


그림 3 - VPL 다이어그램

시뮬레이션 로봇과 연결하기 위해서는 올바른 명령을 Generic Differential Drive 에 설정해 주어야 합니다. 만약 여러 개의 Generic Differential Drive 가 있다면 하나에만 설정해주면 됩니다. 오른쪽 버튼으로 클릭한 후 **Set Configuration** 을 선택합니다. 드롭다운 리스트에서 **Use a manifest** 를 선택, **Import Manifest** 누른 후 아래의 항목 중 하나를 고르면 됩니다.

LEGO.NXT.Tribot.Simulation.manifest.xml,

MobileRobots.P3DX.Simulation.manifest.xml 또는

SimulationTutorial3.manifest.xml.

Run 메뉴 또는 **F5** 를 사용해서 실행 시킵니다. 저장을 하지 않았다면 저장 다이얼로그가 뜨고 이름을 정해서 **Save** 를 눌러 저장하면 됩니다.

VPL 은 이제 애플리케이션을 실행시킵니다. Unblock 을 하겠냐고 묻는 메시지가 뜬다면 **unblock** 을 누르십시오.

시뮬레이션 윈도우가 보일 것입니다.만약 Directional 다이얼로그를 사용했다면 그것도 같이 보일 것입니다. 이제 어떤 다이어그램을 만들었는지에 따라 Xbox 의 Controller 를 사용하던가 다이얼로그 상자의 방향키를 사용해서 로봇을 조종하면 됩니다.

애플리케이션을 멈추기 위해서는 Run 다이얼로그에 있는 Stop 버튼을 누르시면 됩니다.

5. VPL 기본 로보틱스 튜토리얼

로보틱스 튜토리얼 1 (VPL) - 서비스 접근하기

이번 튜토리얼에서는 VPL을 실행하여 샘플 센서를 통해 서비스에 접근하는 과정을 보여줍니다.

시작하기

Visual Programming Langue 개발 환경을 실행한 후 New 메뉴를 클릭하여 새로운 프로젝트를 생성합니다.

Step 1: New Activity 추가 및 구성

서비스 툴박스에서 Generic Contact Sensors service를 끌어다 놓으면 GenericContactSensors 이름의 Activity 블록이 보여집니다.

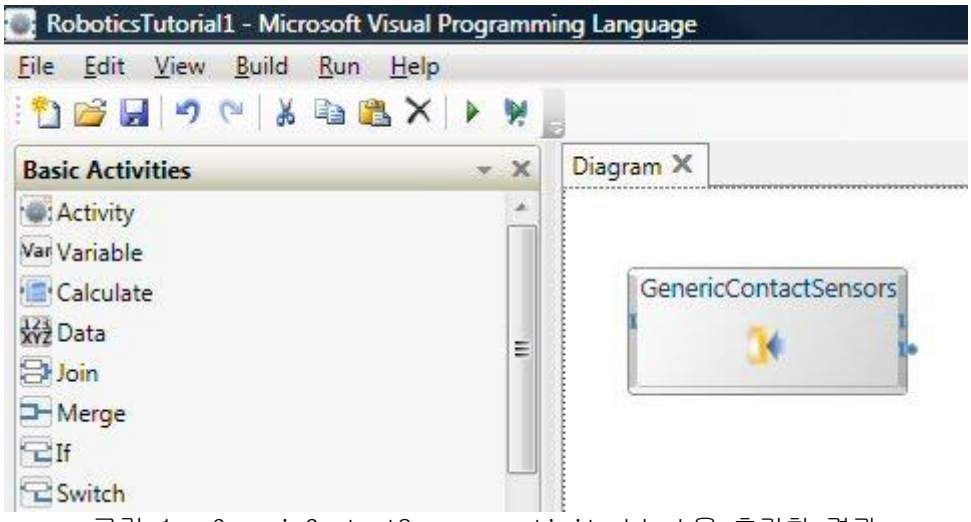


그림 1 - GenericContactSensors activity block을 추가한 결과

Data Activity를 Basic Activities 툴박스에서 선택하여 센서 서비스의 우측에 끌어다 놓은 후 드롭다운 박스에서 String으로 선택한 후 “Ouch!” 라는 문자열을 입력합니다.

Step 2: Activity 연결하기

점으로 표시되어 있는 ContactSensor의 Notification output 부분을 마우스로 끌어서 Data activity에 연결합니다. 정상적으로 연결되면 아래와 같은 다이얼로그 창이 실행되는 것을 볼 수 있습니다. Update와 Create를 선택하고 OK 버튼을 클릭합니다.

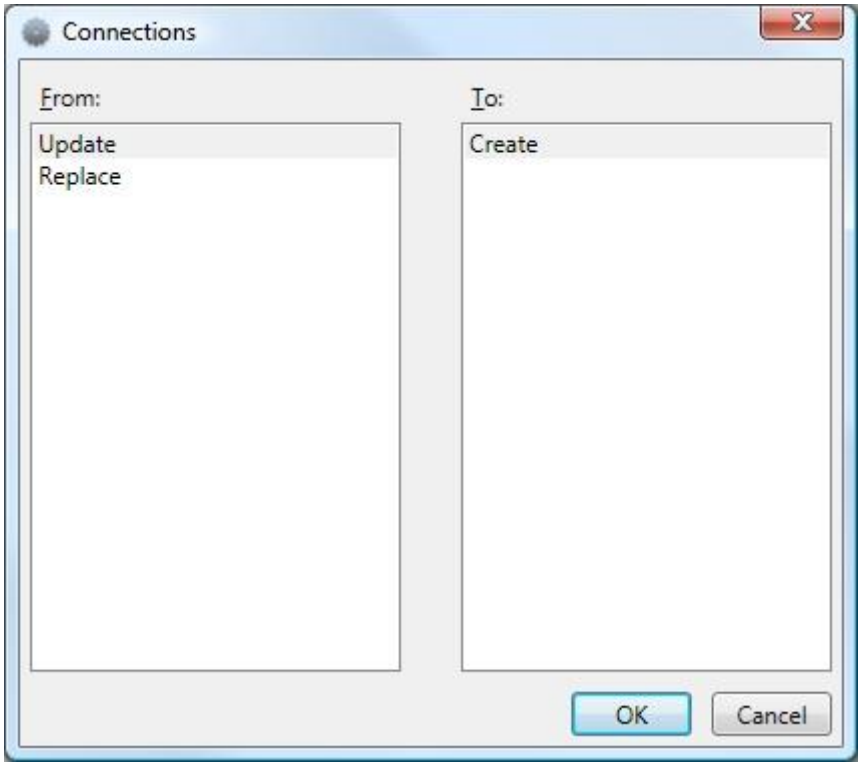


그림 2 - Connections 다이얼로그 박스

이 연결을 통해서, ContactSensor의 Update 메시지가 Data activity의 Create 작업으로 전송되며, 센서로부터 메시지를 받은 후에는 ("Ouch!") 문자열을 생성합니다.

그림 3 - 연결된 Sensor와 Data Activity

Step 3: Simple Dialog Box 추가 및 구성

왼쪽의 서비스 목록에서 Simple Dialog service 마우스로 끌어다가 Data block의 오른쪽에 놓습니다. Data와 Simple Dialog 서비스를 연결하면 다이얼로그 창이 보여집니다. 화면에서 DataValue와 AlertDialog를 선택한 후 OK 버튼을 클릭합니다. 그리고 그 다음으로 Data Connections dialog box에서 Value를 아래 그림과 같이 Value 항목으로 선택합니다.

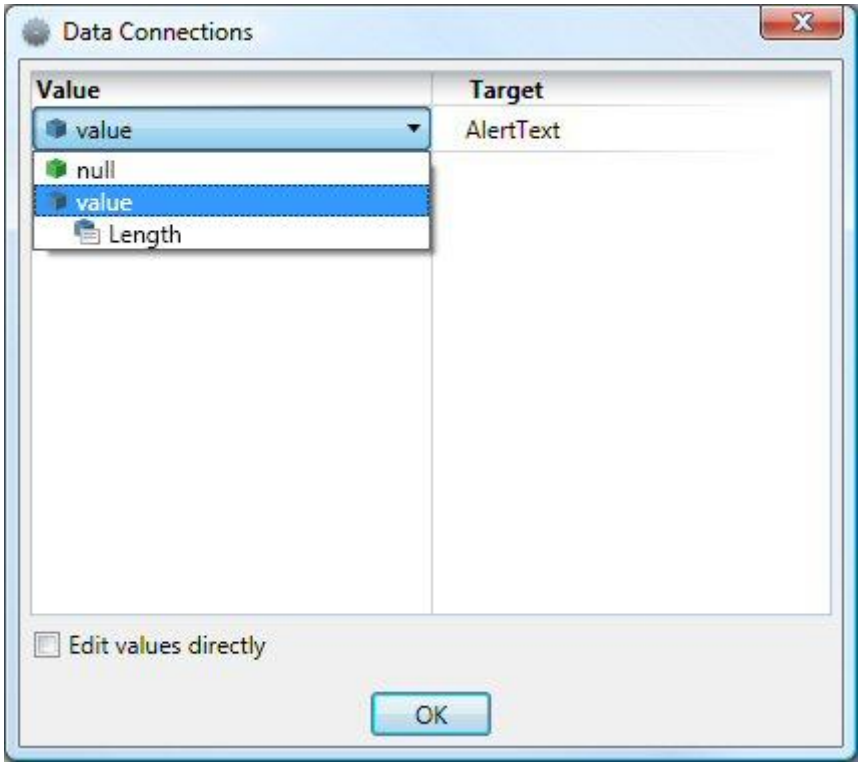


그림 4 - Data Connections Dialog Box

위의 연결을 마치고 나면 전체적인 연결은 아래와 같습니다.

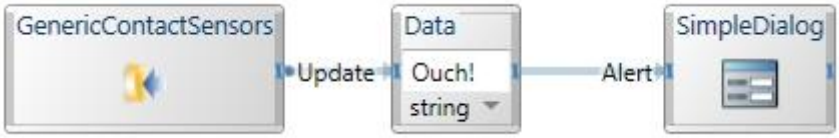


그림 5 - 연결된 Activity들과 Simple Dialog

Step 4: Sensor 지정하기

ContactSensor 블록에서 오른쪽 마우스를 클릭한 후 팝업 메뉴를 실행합니다. 팝업 메뉴에서 Configuration 명령을 선택합니다. 다이얼로그 창에서 Set Configuration 아래에 있는 Use a manifest 항목을 선택합니다. 아무런 매니페스트 항목이 보이지 않는다면, Import Manifest를 선택한 후 해당 목록에서 센서 서비스에 해당하는 매니페스트 파일을 선택합니다.

Step 5: 애플리케이션 실행

메뉴에서 Run 메뉴를 클릭하거나 F5 키를 눌러서 애플리케이션을 실행시킵니다.

로보틱스 튜토리얼 2 (VPL) - 서비스 제어하기

이번 튜토리얼에서는 이전 튜토리얼에 추가하여, Motor 서비스를 추가하고 제어하는 과정을 소개합니다

시작하기

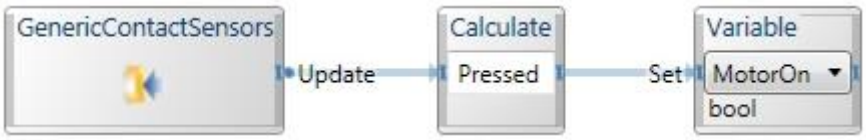
VPL을 실행한 후에 File 메뉴에서 New 항목을 클릭합니다.

Step 1: New Activity의 추가, 구성 및 연결하기

Generic Contact Sensor Array service와 Calculate activity를 추가합니다. 두 개를 연결한 후 Connections 창에서 Update와 Calculate를 connection으로 연결합니다. Calculate text box에 "Pressed" 라고 입력을 합니다.

Step 2: Variable 추가 및 초기화

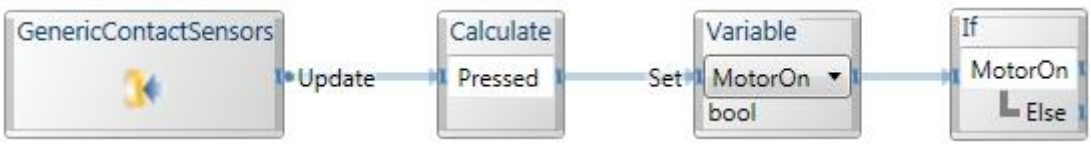
Variable activity를 추가하고 드롭다운 리스트에서 bool 타입의 "MotorOn" 이름을 지정합니다. Variable activity을 MotorOn variable의 Set 형식으로 연결합니다. 또한 CalculatedResult를 Set Value에 연결합니다.



MotorOn variable을 초기화 하기 위해서 또 다른 Variable activity block과 Data activity를 추가합니다. 이 들 두개의 추가된 Activity들에 대해 아래와 같이 설정합니다. 이 들 연결은 별도로 독립적으로 존재하며, MotorOn 값을 초기화 하기 위해 사용됩니다.

Step 3: If Activity 추가

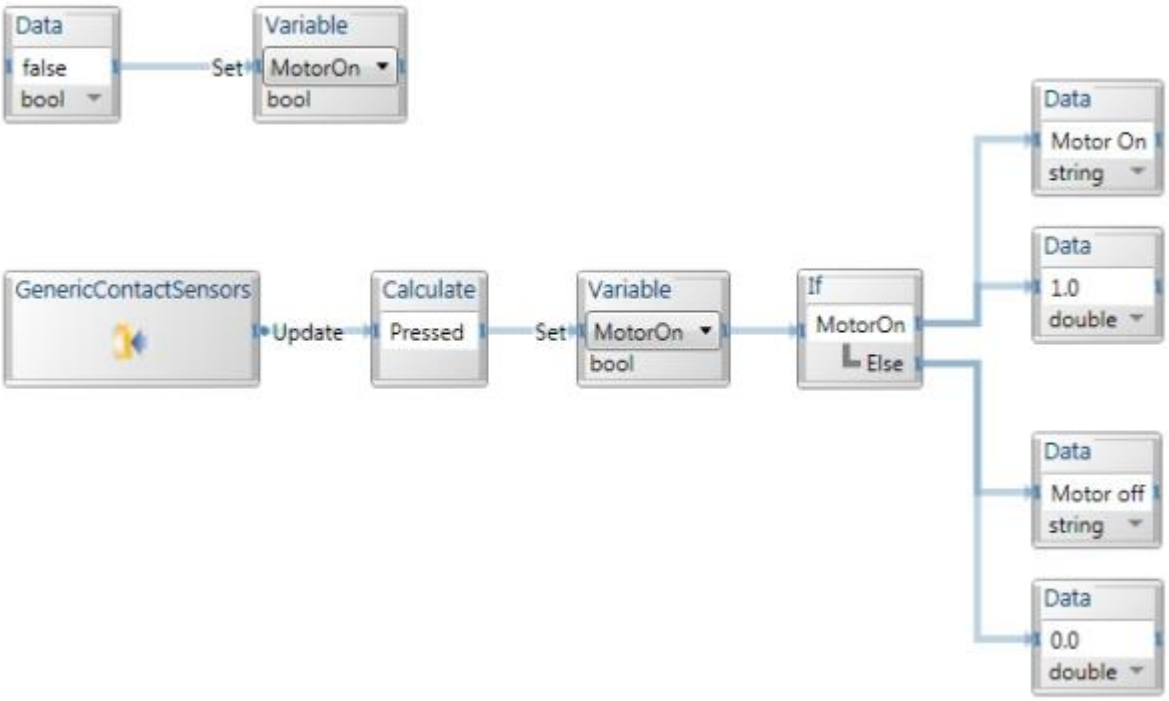
이제 If activity를 추가하고 첫 번째 Variable activity 오른쪽에 위치시킵니다. 그리고 Variable block의 출력값을 If activity의 입력값으로 연결합니다.



여러 개의 Data activity들을 추가한 후에, 첫번 째 항목에는 Sring 타입의 Motor On, 두번째 항

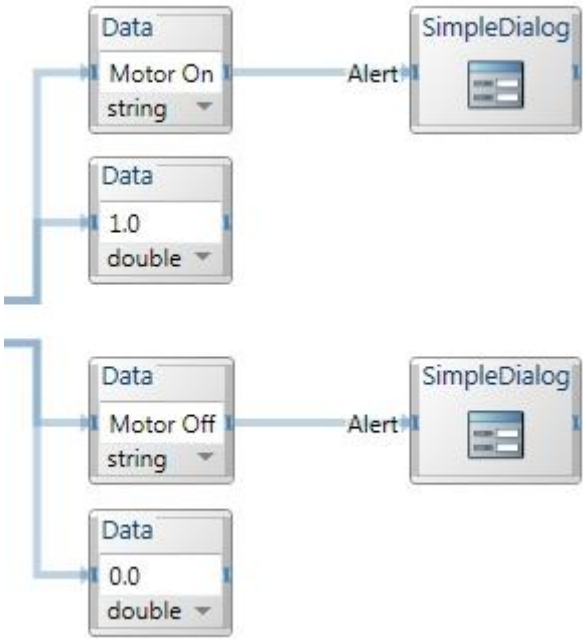
목에는 double 타입의 1.0, 세번째 항목에는 String 타입의 Motor Off, 그리고 네번째 항목에는 double 타입의 0.0 값을 지정합니다.

첫번째 두 개의 항목들은 If의 True 부분에 연결하고 나머지 두 개 항목은 False 항목에 연결합니다.



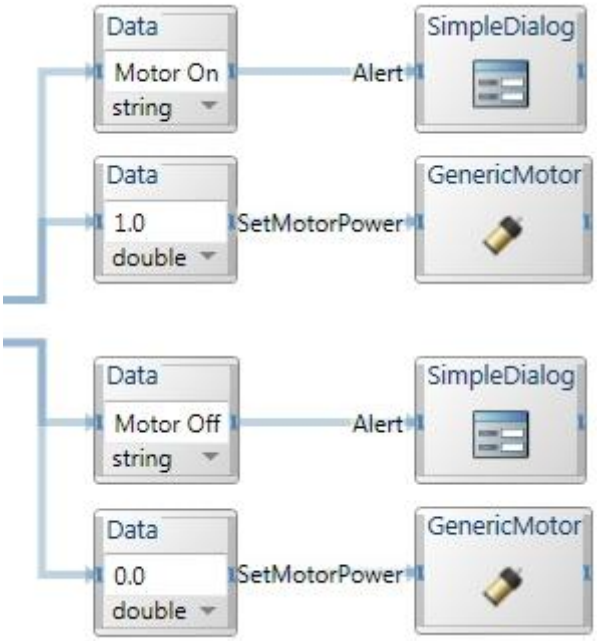
Step 4: Simple Dialog Box 와 Generic Motor 서비스 추가 및 구성

이제 두 개의 Simple Dialog service를 추가하고 각각 MotorOn Data block과 MotorOff Data block에 연결합니다.



이제 Generic Motor service를 추가하고 1.0 값으로 지정된 Data block에 연결합니다.

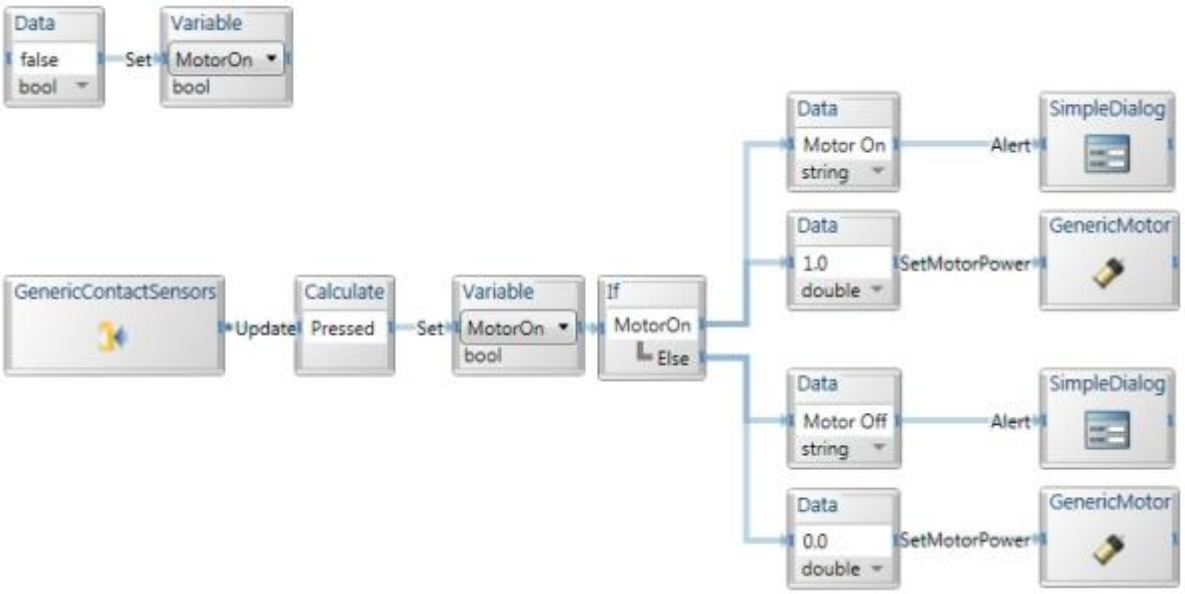
또 다른 Generic Motor를 추가하거나 기존의 서비스를 복사합니다. 이 때 기존의 것을 사용할 것인지 아니면 새로 생성할 것인지를 선택해야 하며, 이 경우에서는 Use Existing option 항목을 선택하고 OK 버튼을 클릭합니다.



Generic Contact Sensor Array service에서와 같이 GenericMotor service에서도 여러 형태의 로봇을 지정할 수 있으며, Motor 서비스를 지원한 매니페스트 파일을 저정하여 해당 로봇과 연결할 수 있습니다.

주의 사항:

두 개의 GenericMotor 서비스가 동일한 모터 서비스를 사용하기 때문에, 두 개 중에 한 개의 서비스에만 구성을 하면 나머지 서비스도 자동으로 설정됩니다.



Step 5: 애플리케이션 실행

메뉴에서 Run 항목을 선택하거나 F5 키를 눌러 해당 프로그램을 실행시킵니다.

로보틱스 튜토리얼 3 (VPL) - 재사용 가능한 오케스트레이션 서비스 생성하기

이번 튜토리얼에서는 자주 사용되는 기능들을 별도우 플로우로 분리하여 생성함으로써, 해당 플로우를 재사용하는 과정을 보여줍니다.

시작하기

VPL을 실행한 후에 File 메뉴에서 New 항목을 클릭합니다.

Step 1: 개발자 자신만의 Activity 생성하기

Basic Activities 항목 중에서 Activity를 선택하여 추가한 후 이름을 RandomDrive로 지정합니다.

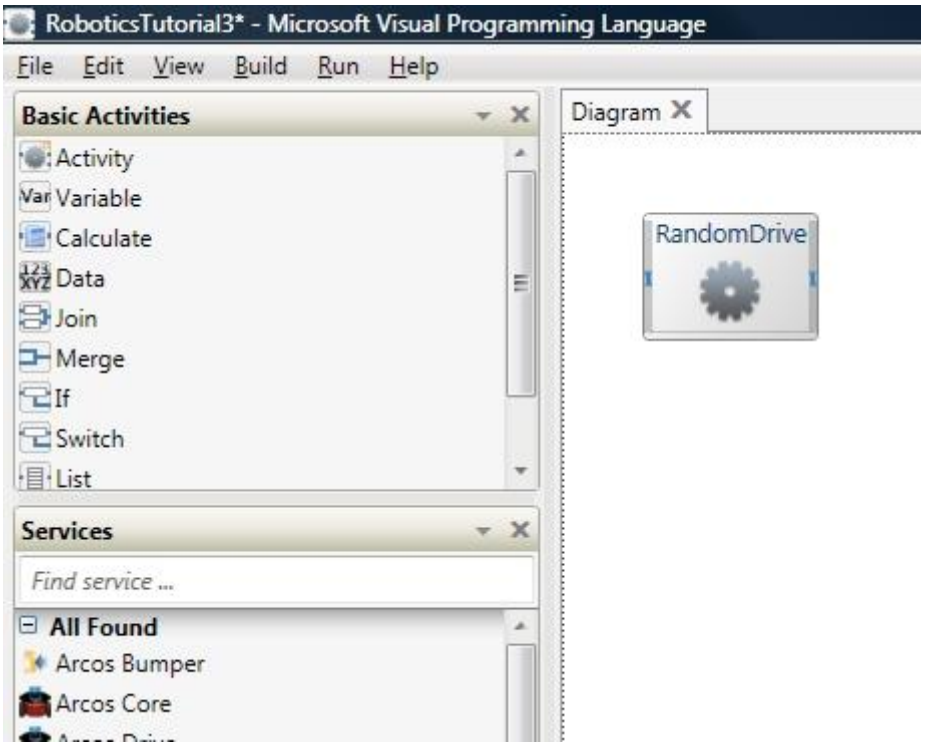


그림 1 - Activity 추가 화면

RandomDrive Activity를 더블클릭하게 되면 아래 그림과 같이 새로운 Diagram 창이 만들어 지는 것을 볼 수 있습니다.

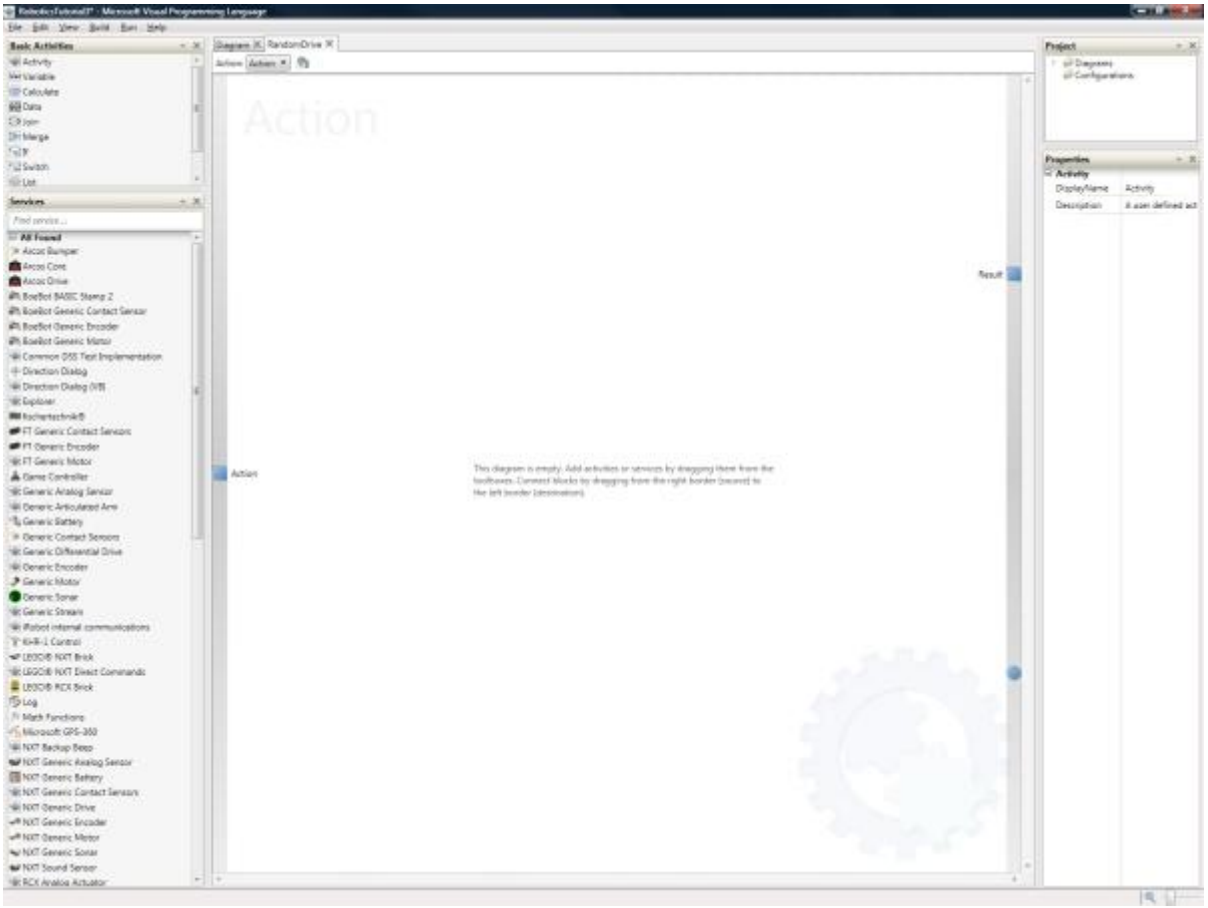


그림 2 - 새로운 activity page

화면 상단을 살펴 보면, “Action:” 항목을 확인할 수 있으며, Start와 Action 값을 선택할 수 있습니다. 이 부분을 Start로 설정할 경우 해당 Activity는 애플리케이션 실행 후 자동으로 실행합니다. 그러나 Action으로 설정하면, 임의의 입력 값을 받아서 기능을 수행하고 결과를 리턴하는 형태가 됩니다. 이 부분을 Action으로 변경해 주고, 우측의 아이콘을 클릭하면 Action과 Notification을 설정하는 다이얼로그 창이 실행됩니다. 다이얼로그 창에서 다음과 같이 Backup , Turn, 그리고 Drive 이름으로 3개의 Action들을 추가합니다.

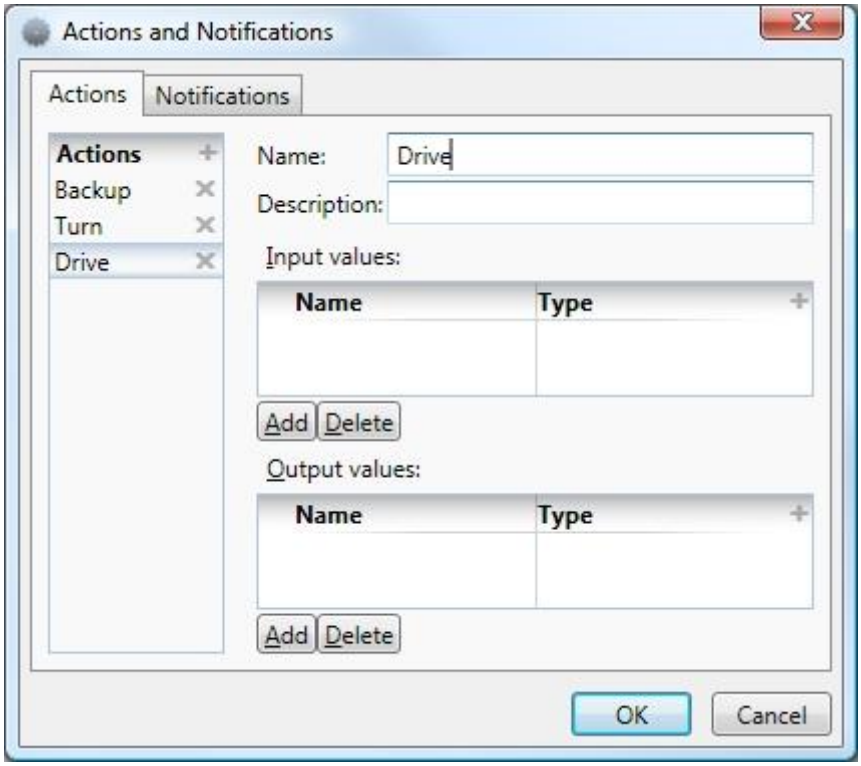


그림 3 - Action과 Notification 다이얼로그 창

위의 3개 Action 항목들에 대해 각각 Input 값을 추가한 다음 이름을 Polarity로 지정하고 double 타입으로 지정합니다.

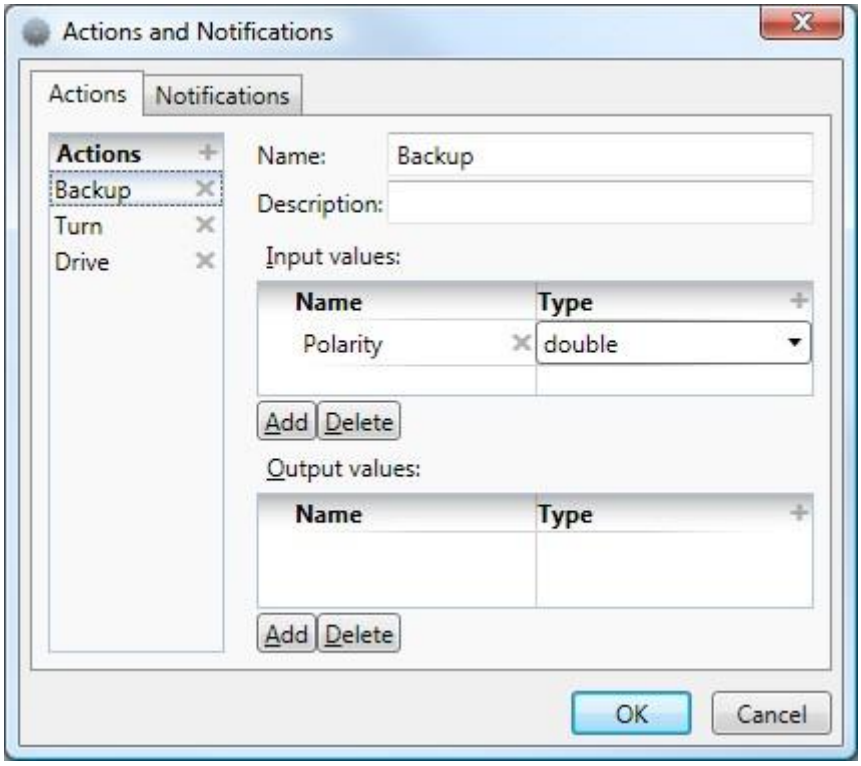


그림 4 - 입력 값 설정

위의 Action 항목 중에서 일단, Turn Action을 선택한 후 OK 버튼을 눌러 창을 닫습니다. 좌측 하단에 있는 서비스 목록 중에서 MathFunction block을 선택하여 추가하고 좌측 상단에 있는 서

비스 중에서 Calculate Activity를 추가한 후에, 좌측에 있는 Input을 나타내는 사각형 점을 MathFunction에 연결합니다. 그리고 MathFunction을 Calculate 항목에 연결합니다.

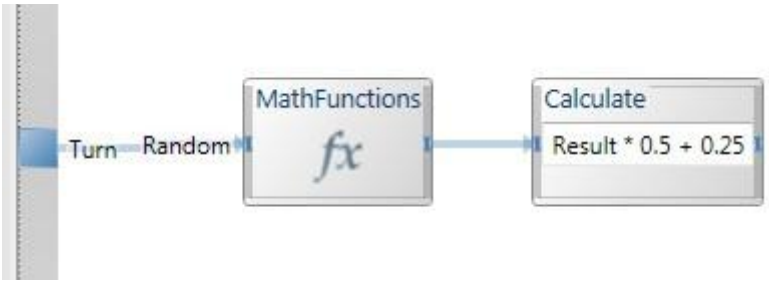


그림 5 - Empty action

MathFunction 목록중에서 Random 항목을 선택합니다. MathFunction block을 Calculate block과 연결할 때에는 Random - Success 항목을 선택합니다. Random 함수는 0.0과 1.0 사이의 렌덤 값을 생성합니다.

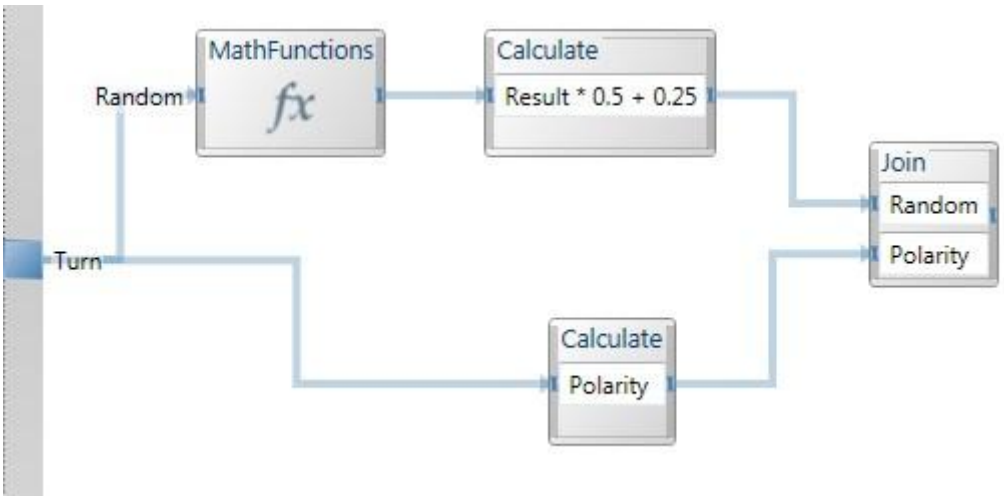


그림6 - Rndom 숫자 생성

위의 연결에서 Random 결과값은 항상 0.25 와 0.75 사이의 값을 가지게 됩니다. 그리고 Input 값에서 Polarity 값을 위와 같이 Calculate를 통해 추출할 수 있으며, 이 값들을 Join Activity를 통해 두 개의 값이 동시에 같이 전달될 수 있도록 합니다.

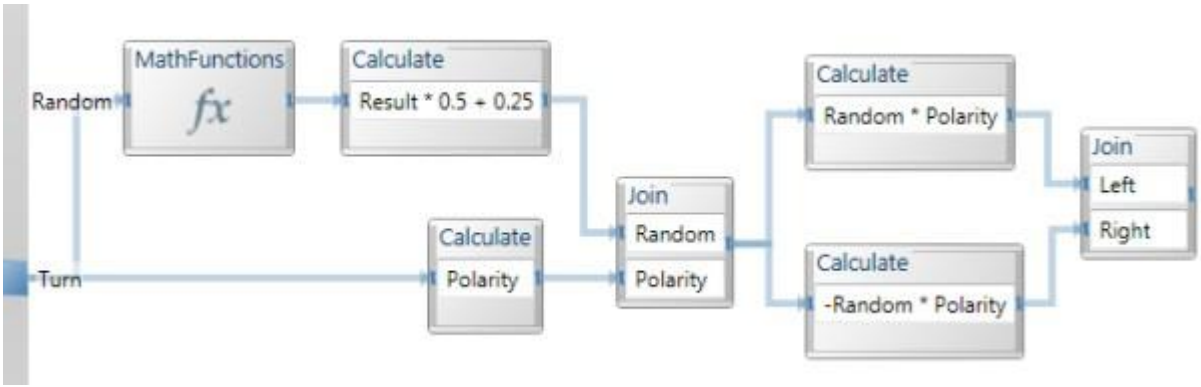


그림 7 - Random 값과 Polarity 값의 Join

로봇의 방향을 변경시키기 위해서는 왼쪽과 오른쪽 모터의 값을 반대로 지정해야 하기 때문에, 위와 같이 연결을 구성합니다.

위와 같이 연결을 마친 후에 Generic Differential Drive Activity를 추가하고 아래와 같이 연결합니다.

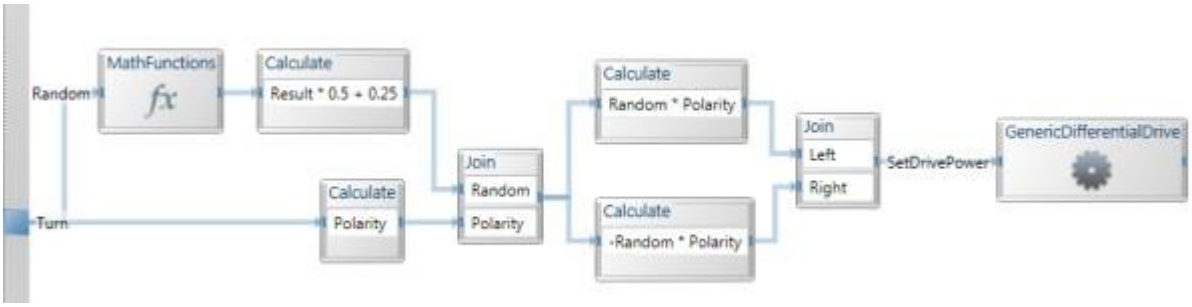


그림 8 - Left와 Right speed 계산

Generic Differential Drive에서 연결시 SetDrivePower 명령어를 선택합니다. 그리고 Data Connections 다이얼로그 창에서 LeftWheelPower에 대해서는 Left, RightWheelPower에 대해서는 Right 값을 매핑합니다.

그림 9 - Unconnected timer

이제 연결된 명령들을 통해 로봇이 방향을 바꾸게 됩니다. 이러한 동작이 특정 시간 후에 발생하기를 원한다면, 위의 그림에서와 같이 타이머를 추가하여 일정 시간을 기다리게 할 수 가 있습니다. 위의 그림과 같이 Data와 Timer Block을 추가하고 1.5초 동안 기다리도록 설정을 합니다.

아래 그림은 모든 Activity들을 연결한 결과를 보여줍니다.

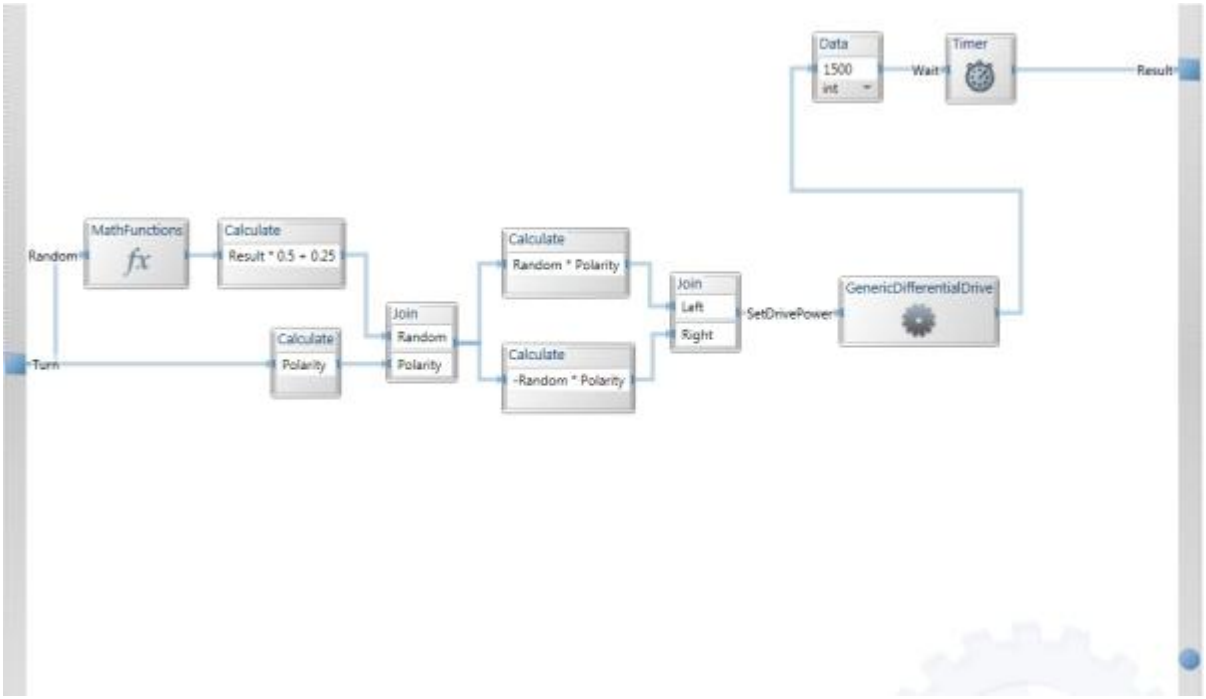


그림10 - 완성된 헨들러

Turn외에 Backup과 Drive에 대해서도 위와 같은 작업을 통해 로봇을 제어하는 모듈을 추가시킬 수 있습니다. 이 때 GenericDifferentialDrive를 추가할 때, 이미 등록된 것을 사용할 것인지 아니면, 새로운 Activity로서 추가할 것인지를 선택해야 하는데, 반드시 아래와 같이 선택해야 합니다.

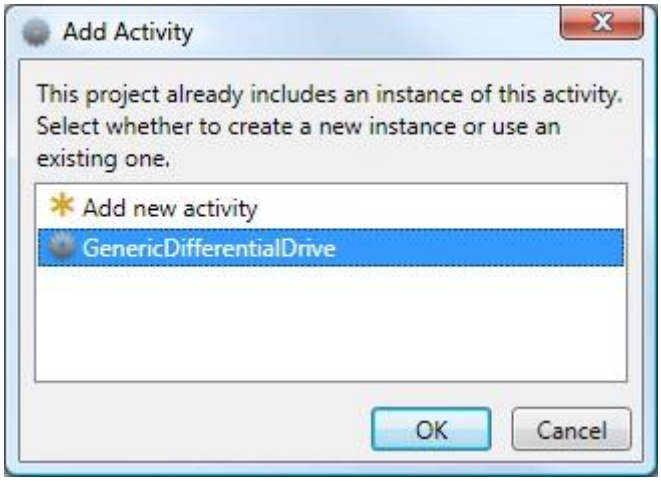


그림 11 - Activity 다이얼로그

Backup action은 Turn과 동일하나 Random 값 대신에 양쪽 Wheel에 동일한 값을 지정합니다.

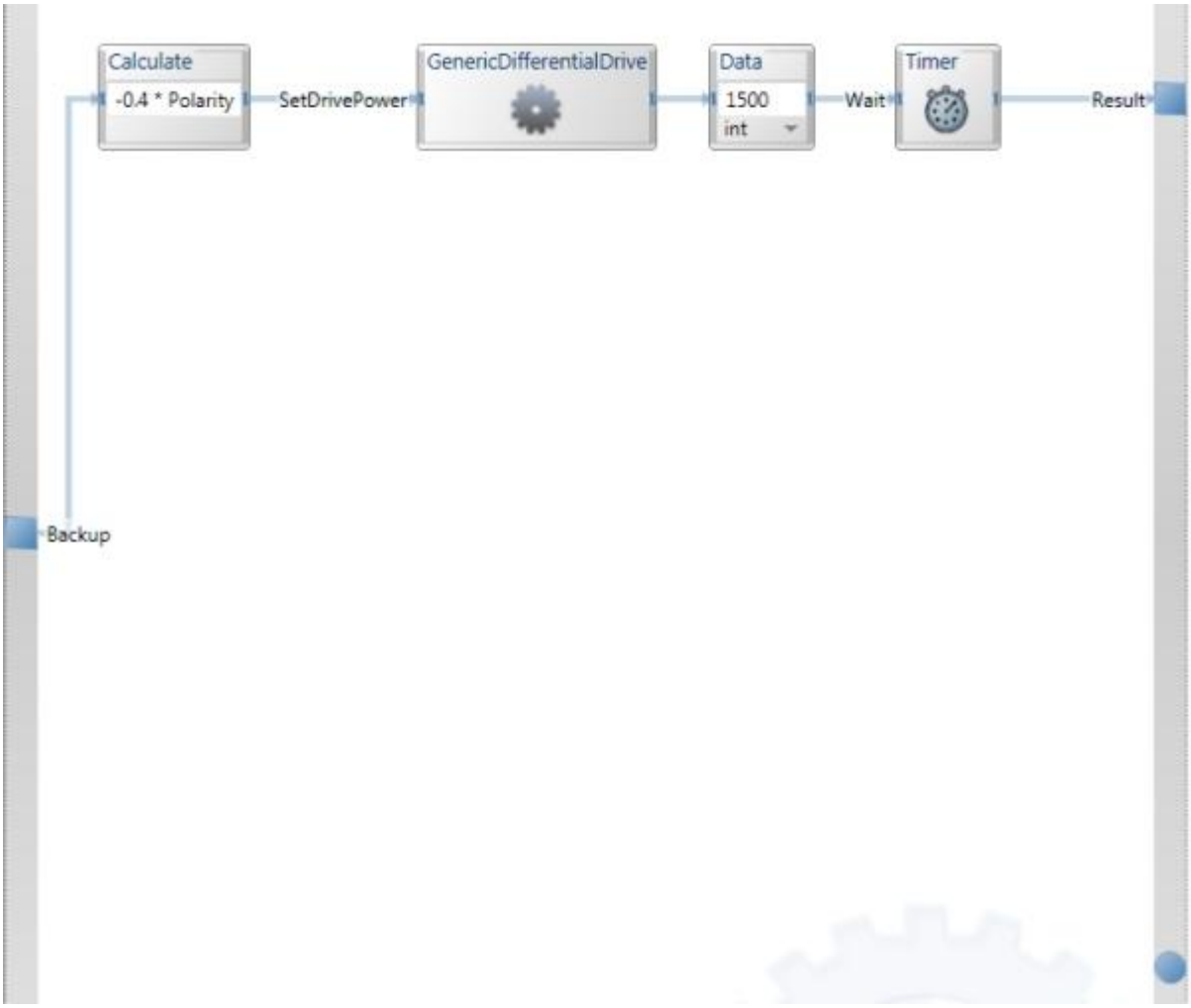


그림 12 - Backup

Drive는 양쪽 wheels에 동일한 Random 값을 사용하며, 타이머를 통해 기다리지 않습니다.

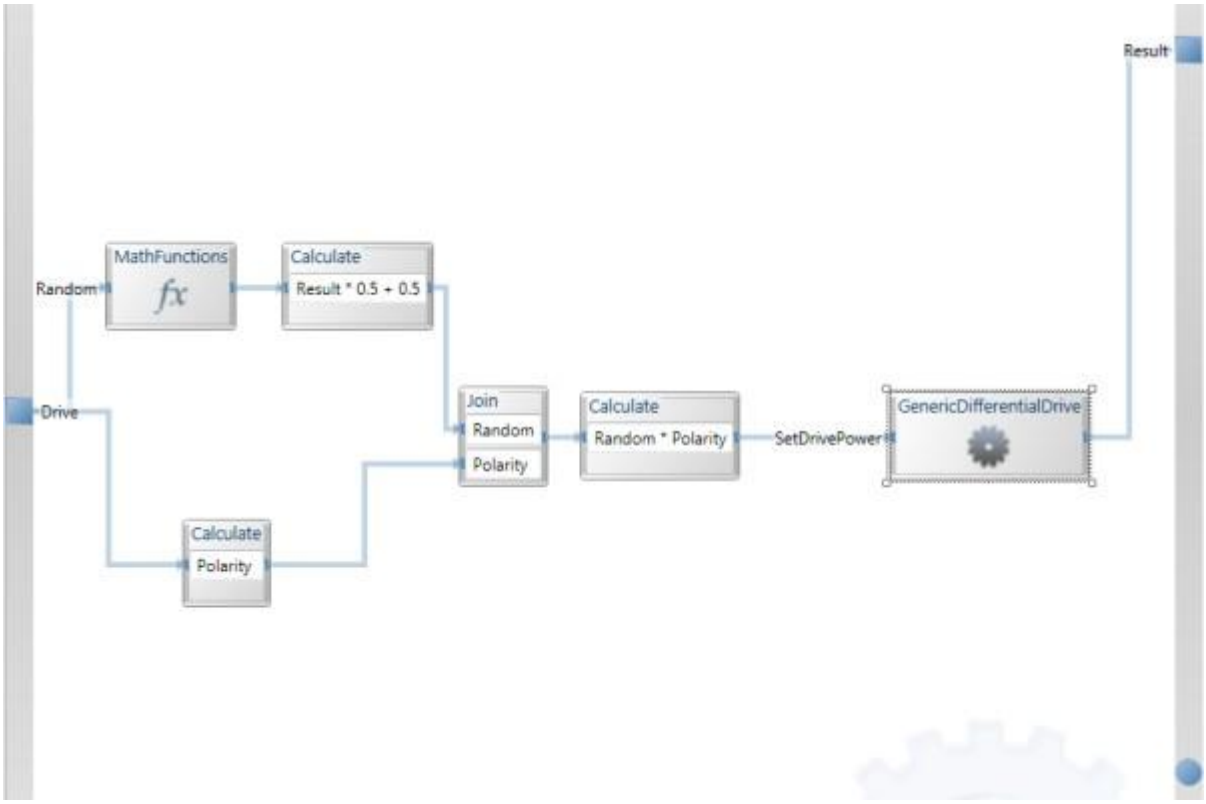


그림 13 - Drive

Step 2: 자체 개발한 Activity 사용하기

이제 지금까지 개발된 Activity를 이전 튜토리얼에서 작업했던 부분에 연결하도록 합니다. 이러한 작업을 위해서는 이전 항목에서의 Variable 부분에 Polarity 이름으로 변경합니다.

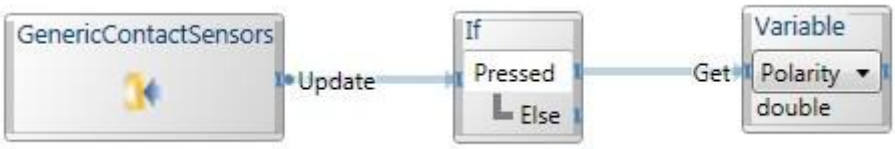


그림 14 - Polarity variable 추가

이제 Polarity 값을 RandomDrive block의 Backup에 아래와 같이 연결합니다.

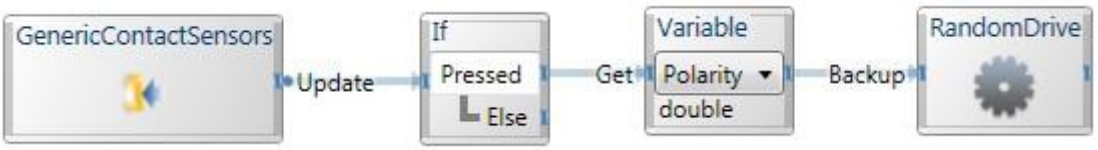


그림 15 - Polarity와 RandomDrive 연결

Backup 후에는 Trun과 Drive에 대해 아래와 같이 연결합니다.

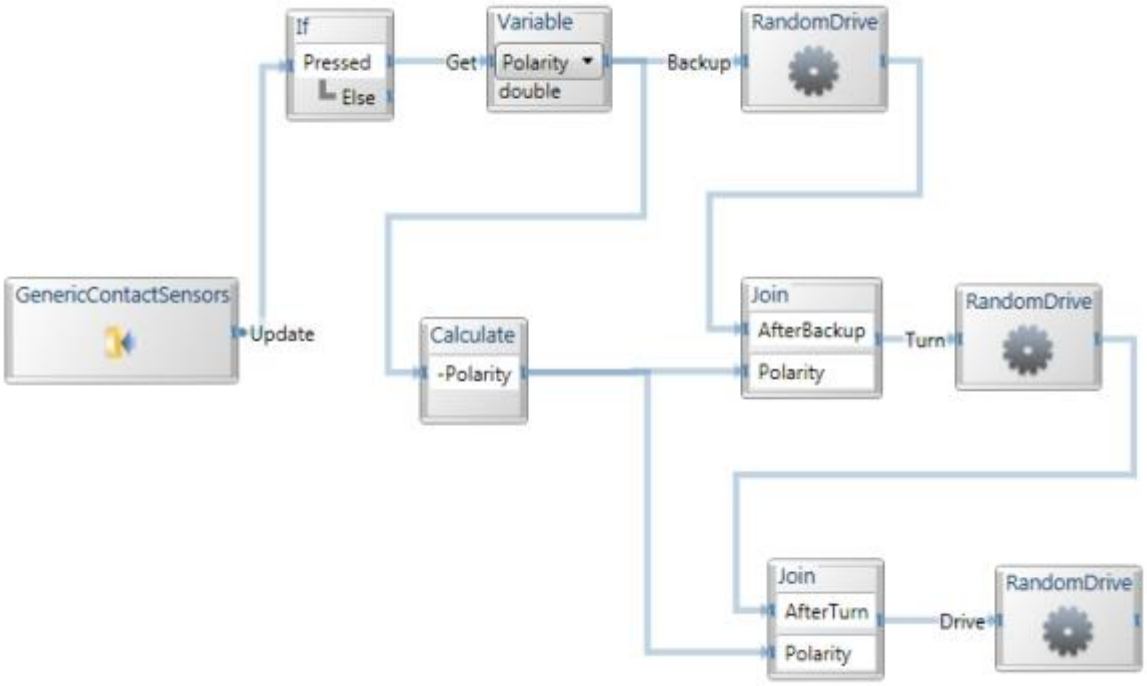


그림 16 - Turn

완성된 다이어그램은 Wander 기능을 수행하며, 아래와 같이 표현될 수 있습니다.

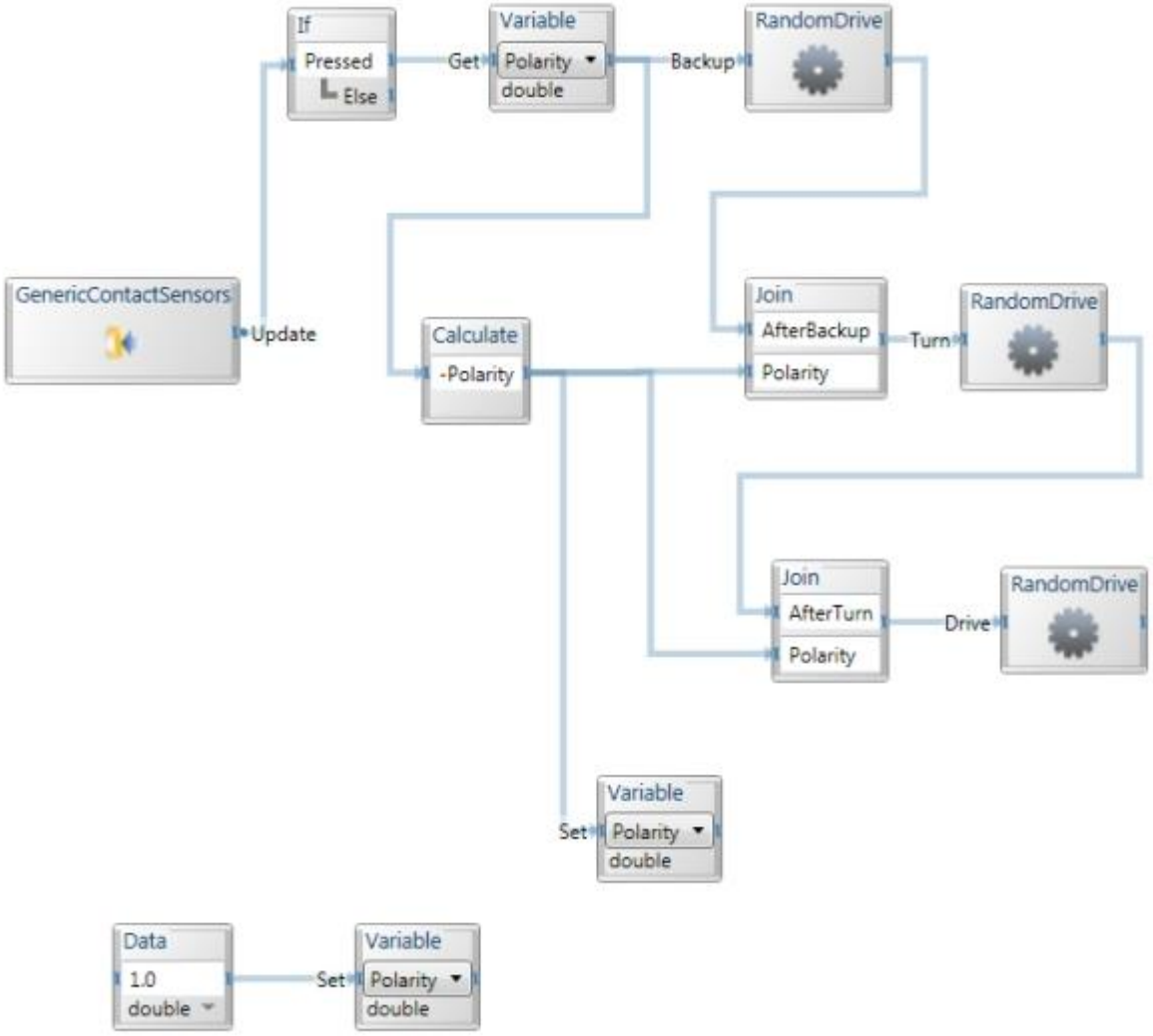


그림 17 - 완성된 wander 다이어그램

Step 3: Define a Manifest for Your Hardware

이제 Sensor와 Motor에 대해 각각 해당 매니페스트 파일을 지정합니다.

Step 4: 애플리케이션 실행

메뉴에서 Run 항목을 선택하거나 F5 키를 눌러 해당 프로그램을 실행시킵니다.

로보틱스 튜토리얼 4 (VPL) - 조종 UI를 통한 로봇 제어

이번 튜토리얼에서는 UI 형태의 로봇 조종 화면을 통하여 로봇을 제어하는 과정을 보여줍니다.

시작하기

Visual Programming Language를 실행한 후에 파일에서 New 항목을 선택합니다. 서비스 항목 중에서 Direction Dialog 끌어다 추가합니다. Direction Dialog는 로봇을 제어하기 위한 명령어를 방향 이름으로 생성을 합니다.

Step 1: 연산 정의

DirectionalDialogService 항목 외에 Calculate Block과 Switch 항목을 등록한 후 아래와 같이 연결합니다. Switch 항목에서 하단의 + 아이콘을 클릭하여 분기 조건을 추가할 수 있습니다.

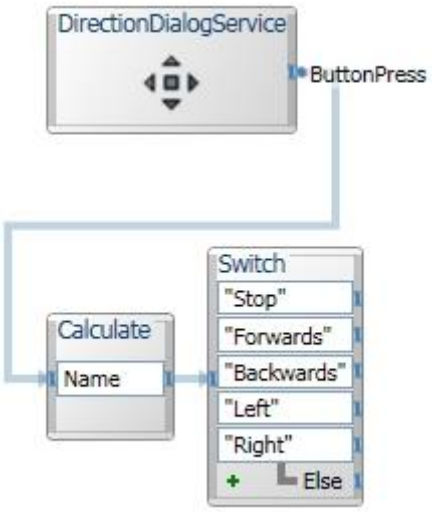


그림 1 - Switch block 연결

그리고 각각의 분기 조건에 대해 Data 항목을 연결시키고 아래와 같이 값을 지정합니다.

"Stop"에는 0.0 입력
"Forwards"에는 0.8 입력
"Backwards" 에는 -0.8 입력
"Left"에는 두 개를 추가한 후 각각 0.6 and -0.6 입력
"Right"에는 두 개를 추가한 후 각각 0.6 and -0.6 입력

두 개의 Join activity를 추가한 후 아래와 같이 연결합니다.

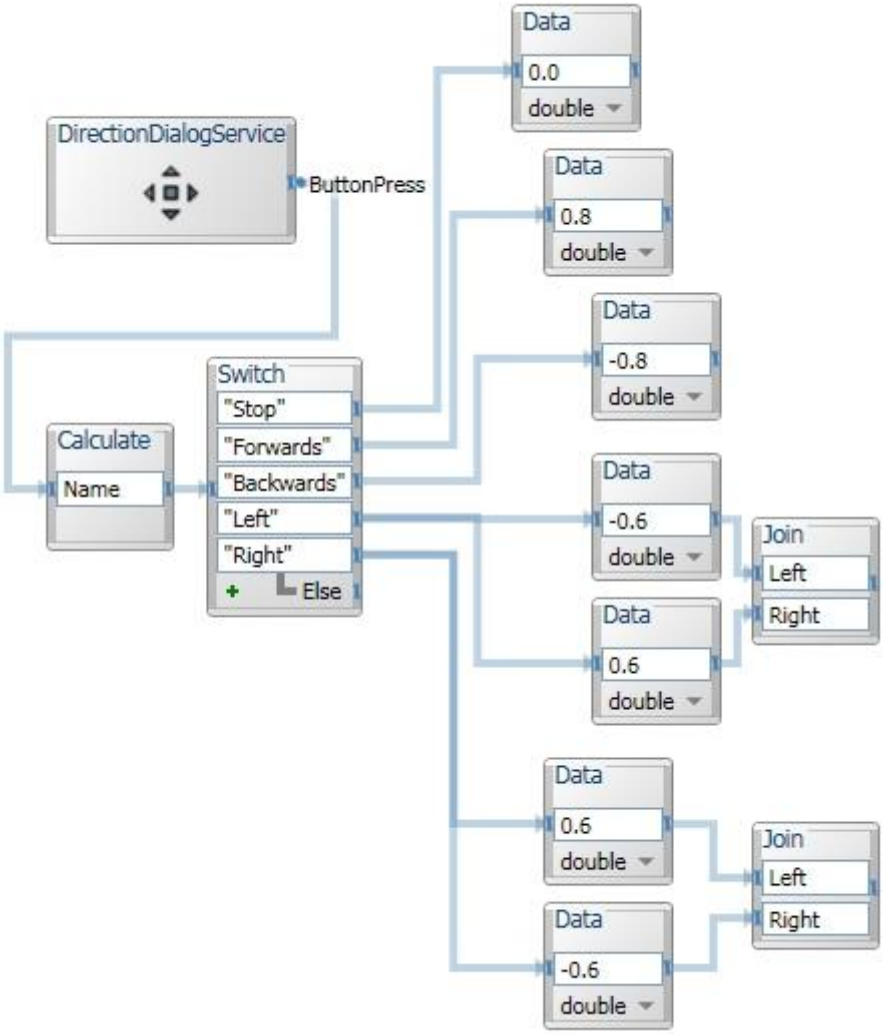


그림 2 - Motor power 값 설정

이제 위의 다이어그램에 Merge Activity와 DriveDifferentialTwoWheel Activity를 추가하여 아래와 같이 다이어그램을 완성합니다.

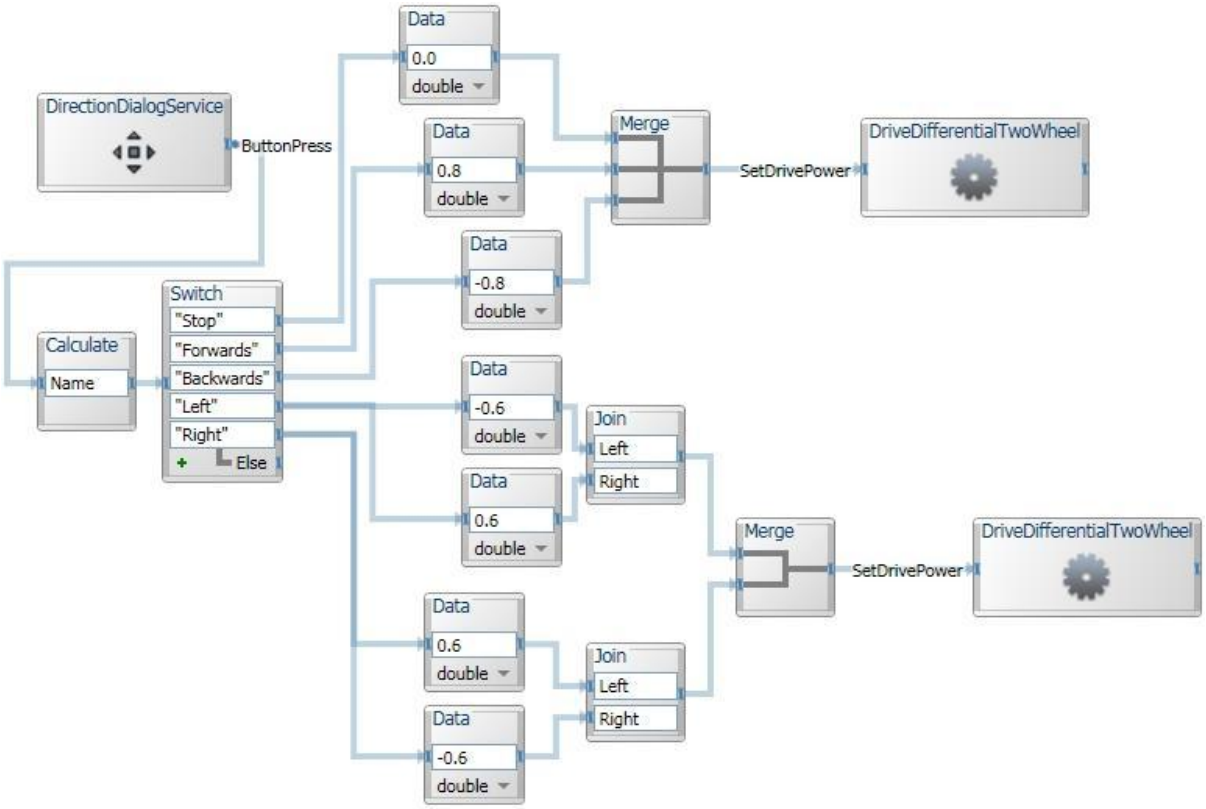


그림 3 - 완성된 다이어그램

Step 2: 애플리케이션 실행

메뉴에서 Run 항목을 선택하거나 F5 키를 눌러 해당 프로그램을 실행시킵니다.