The Qt Company

백서

임베디드 개발자를 위한 Zero Installation 가이드

이 백서는 Zero Installation이 어떻게 오늘날의 모습으로 발전했는지를 살펴봅니다. 또한 이 솔루션의 구성요소를 설명하고, 이를 배포하는 데 사용될 수 있는 세부적인 기술에 대해 알아보며, 차세대 임베디드 제품에 이를 포함시킬 때 고려해야 할 사항에 대해 살펴봅니다.

목차

서론 및 용어

Zero installation은 애플리케이션이 브라우저에 기반하고 있기 때문에 임베디드 장치의 사용자가 애플리케이션을 다운로드하거나 설치하지 않고도 실행할 수 있습니다. 일반적인 설치 프로세스를 없앰으로써 사용자는 이전과는 다른 능률적인 경험을 할 수 있습니다. 또한 브라우저 기술은 기본적으로 표준화되어 있기 때문에 Windows/Mac/Linux 데스크톱, Android, 아이폰이나 다양한 종류의 태블릿을 비롯한 모든 클라이언트 장치에서 하나의 애플리케이션을 실행할 수 있습니다.

Zero Installation은 사용자가 앱을 설치하는 대신 웹 페이지를 이용해 스마트폰, 태블릿 또는 데스크톱 등의 임베디드 장치를 제어할 수 있도록 해 줍니다.

Zero installation은 0Install과 이름은 비슷하지만 이와 같은 특정한 소프트웨어 패키징 및 배포 도구가 아니며, 브라우저 기반 애플리케이션을 지칭하는 일반 용어에 더 가깝습니다. Zero Installation에 대한 정의는 몇 가지가 있겠지만, 이 백서의 목적상 Zero Installation을 장치가 호스팅하는 브라우저 기반의 컴패니언 프로그램을 통해 임베디드 장치를 제어하는 한 가지 방법으로 간주합니다. (백서의 뒷부분에서는 클라우드 호스팅 애플리케이션으로 임베디드 장치를 구동할 수 있는 대안적 Zero-Installation 구성에 대해서도 살펴볼 것입니다.) 정의와 관계없이 모든 Zero Installation 솔루션에는 여러 가지 구성 요소가 있습니다.

임베디드 장치 – 스마트 홈 제어센터, 산업용 로봇, X선 장치에 관계 없이 Zero Installation 솔루션에는 항상 하드웨어가 있습니다. 임베디드 장치는 장치를 제어하는 방법에 대해 사용자에게 알려 주어야 하며, 사용자가 그 상태를 확인할 수 있도록 해야 합니다.

연결 – 임베디드 장치는 WiFi, 이더넷 또는 이동통신을 통해 TCP/IP 연결과 같은 몇 가지 형태의 연결을 제공해야 합니다. IoT 장치의 경우 완벽한 인터넷 연결을 제공할 수 있으나 이것이 유일한 방법은 아닙니다. 인터넷을 거치지 않는 전용 또는 직접 네트워크 연결이 또 다른 대안으로, 보안이나 기능 안전성이 중요한 분야에서는 더 바람직할 수 있습니다.

사용자 장치(또는 클라이언트) – 원격 사용자 인터페이스를 표시하기 위해서는 데스크톱 컴퓨터, 노트북, 스마트폰 또는 태블릿을 비롯한 클라이언트 장치가 필요합니다. 이러한 클라이언트 장치에서는 원격 애플리케이션을 로드할 수 있도록 충분한 여유의 메모리로 최신 브라우저를 실행할 수 있어야 합니다.

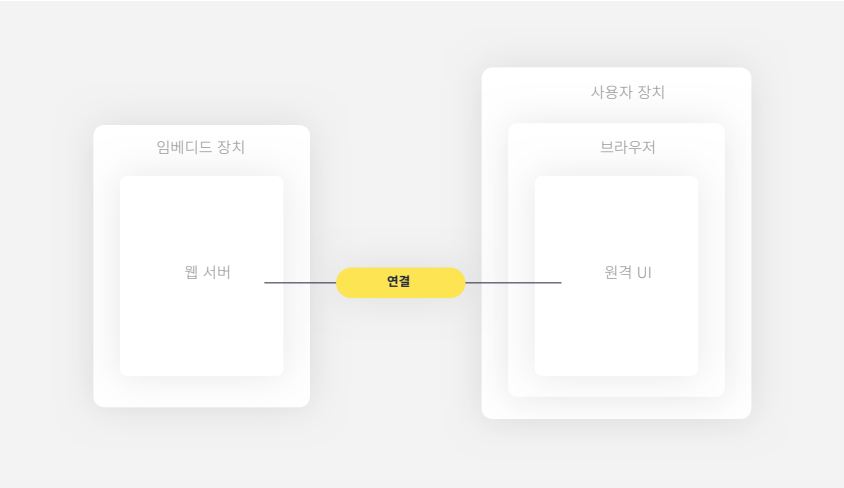
원격 UI(또는 원격 애플리케이션, 또는 컴패니언 앱) – 임베디드 장치를 제어하려면 사용자의 브라우저 안에서 소프트웨어가 실행되어야 합니다. 물리적 디스플레이와 버튼이 있을 수 있지만 이 UI가 장치의 주 사용자 인터페이스가 되는 경우가 종종 있습니다.

브라우저 – 애플리케이션이 실행될 가상 환경을 제공하기 위해서는 클라이언트 장치에서 실행되는 웹 브라우저가 있어야 합니다.

웹 서버 – 브라우저에 애플리케이션을 제공하기 위한 소프트웨어가 임베디드 장치에 필요합니다.

현실에서의 Zero Installation

Zero installation 기술은 제조업체뿐만 아니라 사용자에게도 많은 이점을 제공합니다. 이러한 이유로 이 기술은 여러 다양한 분야에 성공적으로 적용되고 있습니다.



Zero installation의 핵심 구성요소

산업 자동화 – 여러 기업들이 공정 제어 또는 자동화 제어 소프트웨어에 Zero Installation을 활용하고 있습니다. Bosch가 좋은 예입니다. 이 기업은 조립라인 보정 장비에 이를 활용하고 있습니다.

자동차 – 차량 모듈 내의 Zero installation 기능을 통해 딜러와 정비사가 쉽게 문제를 진단할 수 있습니다.

의료 – 다수의 선도기업이 의료 영상 확인, 환자 기록 관리, 진단 공유에 Zero Installation 솔루션을 활용하고 있습니다.

스마트 홈 – 기술자들은 스마트 홈 허브를 통해 사용자 앱보다 한 차원 높은 고급 UI를 활용하여 홈 시스템을 연결하고 구성할 수 있습니다.

사용자(또는 운영자)와 기술자가 모니터링, 제어 또는 진단하는데 필요한 첨단 임베디드 시스템을 탑재한 거의 모든 제품이 Zero Installation를 탑재할 후보입니다. 산업 분야로는 첨단 제조, 항공우주, 농업, 건설, 소비자 가전, 의료, 자원추출, 로보틱스, 운송 등을 들 수 있으며, 거의 전 분야가 포함될 것입니다.

최신 Zero Installation이 차별화되는 이유

새로 산 프린터에 구성 소프트웨어나 드라이버가 담긴 설치 디스크가 들어 있던 시절을 기억하시는 분들도 있을 것입니다. 별도의 설치 단계를 거쳐야 했기 때문에 늘 불편했고, 하드웨어 하나를 여러 명이 공유하는 IT 환경에서는 특히 성가신 일이었습니다. 회사의 네트워크에서 전 직원이 이용하는 소프트웨어를 관리할 경우에 발생하는 큰 불편함을 덜어주기 위해 초기의 Zero Installation 솔루션에는 프린터나 네트워크 스토리지 장치와 같은 공유 장치에 웹 서버와 웹 애플리케이션이 추가되어 있었습니다. 엑스트라 스토리지에 드는 비용 때문에, 웹 페이지 구성에서 다양한 기능은 종종 상대적으로 비싼 장치에만 국한되어 추가되었습니다. 여러 가지 이유로 오늘날의 Zero Installation 솔루션은 이전과 달라졌습니다.

사용자 – IT 전문가만 장치와 소통해야 하는 것은 아닙니다. 장치에 따라 진단 기술자, 특수 교육을 받은 작업자 또는 소비자도 소통을 필요로 합니다.

제품 – Zero Installation은 한때 고급 제품 범주에만 국한되었지만, 이제 연결성은 거의 모든 임베디드 장치에서 보편화되어 있습니다. 이제는 인터넷 연결과 함께 웹 서버를 실행할 정도의 성능을 갖춘 IoT 장치도 똑같이 좋은 제품입니다.

모양과 느낌 – 이전 세대의 임베디드 장치들은 외형이 단순했으며 투박한 UI를 사용했습니다. 이제 웹 기술 표준은 성능도 훨씬 뛰어나고 보기도 좋은 다수의 옵션을 제공합니다.

최종 결과 – 오늘날의 Zero Installation 솔루션은 단순한 구성 또는 모니터링 작업에 집중하기보다는 모든 기능을 갖춘 브라우저 기반의 앱을 제공합니다. 이러한 앱은 정교함과 기능 면에서 데스크톱 애플리케이션과도 비견될 수 있을 정도입니다.

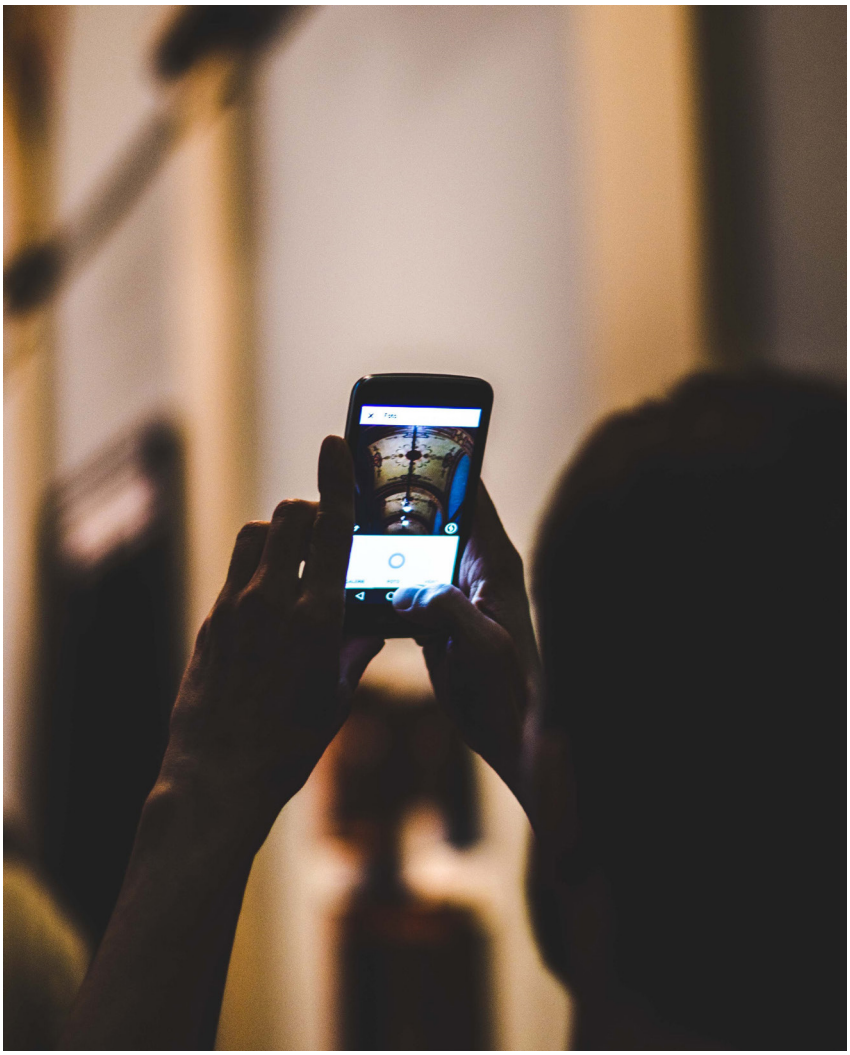
사용자의 이점

오늘날의 Zero Installation 솔루션은 제품 기능에 "즉각적으로" 액세스할 수 있는 방법을 제공하기 때문에 소비자에게 매우 편리합니다. 예전부터 새 제품을 사용할 때 처음 얼마 동안 사용자가 맞닥뜨리게 되는 어려움은 사소한 문제가 아닙니다. 이는 사용자 경험을 부정적으로 만들고, 구매 효용을 크게 떨어뜨릴 수 있으며, 브랜드에 부정적인 영향까지 미칠 수도 있습니다. 뭐든지 "별 탈 없이 작동할 것"이라고 기대하는 것은 인간의 본성입니다. 긍정적인 사용자 경험은 제품의 사용 대상이 소비자인지, 기술자인지와 무관하게 반드시 갖춰야 할 부분입니다.

이전의 Zero Installation 솔루션은 주로 원격 데스크톱에서 장치를 모니터링하는 수단을 제공한다는 점에서 그 가치를 인정 받았습니다. 그러나 이제 사람들은 책상 앞에서만이 아니라 가정이나 길가에서도 자신의 장치를 제어하고 싶어합니다. 소비자가 자신의 모바일 장치를 사용하여 제품을 제어할 수 있도록 함으로써 Zero Installation은 이동의 자유를 선사할 뿐만 아니라 어디에서든 접속할 수 있도록 해 줍니다. 공장의 전용 컴퓨터뿐만 아니라 인터넷 카페에서도 접속이 가능합니다.

사용자가 자산의 장치를 사용해 UI를 제어할 수 있게 되면 추가적인 장비의 필요성도 없어집니다. 제품을 진단, 구성하거나

Zero Installation은 사용자에게 이동의 자유를 선사할 뿐만 아니라 어디에서든 접속할 수 있도록 해 드립니다.



수리하기 위해 전문화된 도구를 필요로 하는 대신, Zero Installation은 사용자가 전문화된 새로운 도구 없이도 제품의 기능에 대한 접속 권한을 얻을 수 있도록 도와줍니다.

제조업체의 이점

장치의 제조사도 자사의 제품에 Zero Installation을 구축해야 할 여러 가지 이유가 있습니다. Zero Installation 앱은 기본적으로 크로스 플랫폼입니다. 이는 Windows, Mac, iOS, Android, Linux 빌드를 별도로 구축하는 대신 제품의 모든 잠재적 사용자를 지원하는 단 하나의 솔루션을 개발할 수 있다는 의미입니다. 오랜 시간이 걸리는 설치 과정을 생략하고 고객이 선택한 플랫폼을 지원함으로써 제품 제조사는 더욱 향상되고 효율적인 경험을 제공하여 자사의 제품을 차별화할 수 있습니다. 또한 공유 라이브러리의 비호환성이나 부적절한 사용자 권한과 같은 설치 시 발생할 수 있는 "에러"도 없습니다.

Zero Installation이 적합하지 않은 경우

Zero Installation으로 사용자 경험의 마찰을 줄이고 더 우수한 제품을 더 쉽게 만들 수 있다면 향후의 시스템에 도입하지 않을 이유가 있을까요? 여기에는 몇 가지 고려할 요인이 있습니다.

Zero Installation은 사용자 경험의 마찰을 줄이고 더 우수한 제품을 더 쉽게 만들 수 있도록 합니다.

독창성 없는 UI
모든 클라이언트에게 하나의 애플리케이션이 제공되므로 사용자 인터페이스도 분명히 모든 장치에서 동일할 것입니다. 이러한 일관성은 많은 경우에 장점으로 작용할 수 있지만 UI에 고유한 모습과 느낌이 없을 수 있습니다. 더욱 중요한 것은 이러한 UI가 브라우저 환경을 벗어난 장치의 고유한 기능을 전혀 통합할 수 없다는 것입니다. 플랫폼을 따르는 UI나 플랫폼 특유의 센서가 제품의 정의에 필수적이라면 Zero Installation 솔루션은 이에 적합하지 않을 수 있습니다. 이러한 경우 여러 플랫폼 버전별로 컴패니언 앱을 따로 개발해야 할 수 있습니다.

사용자간 소통의 부재
센서나 엣지 장치와 같은 몇몇 부류의 장치는 사용자와 직접 소통할 필요가 없습니다. 이러한 소형 장치들은 종종 더 큰 구성요소의 일부가 되어 게이트웨이 장치나 클라우드 애플리케이션에 연결됩니다. 특히 이들은 개별적으로 액세스할 필요가 없기 때문에 UI가 전혀 필요하지 않습니다. 이 장치들이 MQTT 또는 SNMP와 같은 표준 프로토콜을 사용하는 경우에도 마찬가지입니다. 사용자는 이러한 장치들을 해당 프로토콜을 사용하는 기존의 대시보드 애플리케이션에 연결할 수 있기를 기대하므로, 별도의 Zero Installation 앱을 호스팅한다고 해도 추가적인 이점이 없습니다.

TCP/IP 미지원
Zero Installation을 피해야 하는 이유 중 한 가지 중요한 것은 임베디드 장치가 WiFi, 이더넷 또는 이동통신을 지원하지 않을 경우입니다. 블루투스 LE를 사용하는 저전력 장치와 Zigbee 또는 Z-Wave 메시 네트워크를 통해 소통하는 장치가 그러한 예가 될 수 있습니다.

저장소 부족
내장 마이크로 컨트롤러 플래시 메모리를 사용하는 소형 장치나 저가 장치에는 웹 서버나 웹 애플리케이션을 호스팅할 수 있는 저장소가 충분하지 않을 수 있습니다. (장치 자체에 충분한 공간이 없다고 하더라도 정말로 Zero Installation을 설치하고 싶으시다면 클라우드 서버에 앱을 호스팅하는 것이 대안이 될 수 있습니다.)

부족한 연결성
신뢰할 수 있는 연결이 불가능하거나 아예 연결이 제한되어 있는 환경도 있습니다. 그 이유가 인프라의 부재든, 보안 문제든, 도달 범위가 불충분하든 관계 없이 Zero Installation 제품의 배제 여부를 판단 하려면 자신의 장치와 장치가 궁극적으로 설치되는 네트워크 환경을 이해해야 합니다.

제한된 장치 액세스
개발 아키텍처에 따라 Zero Installation UI는 임베디드 시스템의 하드웨어에 직접 액세스하지 못할 수도 있습니다. 이는 UI와 하드웨어를 연결하기 위해서는 추상화 계층이 필요하며, 이에 따라 개발 시간이 늘어나고 더욱 복잡해지며 솔루션에 지연이 발생한다는 것을 의미합니다. (뒤에서 설명하겠지만 Qt WebGL 구현에는 이러한 제한이 없습니다.)

오늘날의 Zero Installation 기술 포트폴리오

종종 암호 같고 다루기 힘든 인터페이스를 제공했던 이전 세대의 임베디드 장치는 이전 버전의 HTTP 및 HTML에만 국한되어 있었습니다. 특히, 대중적이고 유려하며 직관적인 사용자 인터페이스가 사용자의 기대를 한껏 높여 놓은 상황에서 과거의 초보적인 UI는 더 이상 환영 받지 못하고 있습니다. 그 덕분에 이제는 Zero Installation 애플리케이션을 구현하는 데 여러 가지 새로운 기술 옵션을 활용할 수 있게 되었습니다.

HTML5
HTML5 마크업 언어는 특히 파트너인 JavaScript 및 CSS3와 결합되면서 초기 웹 시대 이후 기능 면에서 크게 향상되었습니다. HTML5는 이제 Google Docs, Airtable 이나 Pixlr와 같은 완전 대화형 앱을 지원합니다.

WebGL
기본적으로 그래픽 또는 3D를 표현하기 위한 브라우저 기반 2D/3D 그래픽 API인 WebGL 은 브라우저의 그래픽 컨텐츠 표시를 지원하는 OpenGL ES에 바탕을 두고 있습니다. Web GL 애플리케이션의 예로는 Google Maps, Thingiverse 3D 프린팅 플랫폼, BioDigital Human Platform을 들 수 있습니다.

WebAssembly
Web Assembly 는 일반적으로 브라우저 내에서 실행할 수 없는 언어(C 또는 C++ 등)를 WebAssembly 지원 브라우저와 호환되는 형태로 개발자들이 교차 컴파일할 수 있게 해 줍니다. 현재 Google Chrome, Mozilla Firefox, Opera, Microsoft Edge, Apple Safari를 비롯한 모든 주요 브라우저에서 이를 지원하고 있습니다. WebAssembly는 가상화된(그리고 안전한) 환경에서 작동하지만 고유의 실행 속도에 근접하는 것을 목표로 하고 있습니다. 살펴보실 수 있는 여러 사례를 이 백서에서 소개하고자 합니다.

WebSocket API(웹 소켓)
브라우저 애플리케이션은 HTTP를 통해 임베디드 장치와 통신하는 경우가 많은데, 이는 HTTP와 임베디드 애플리케이션 서버가 애초에 그 설계에 반영되지 않은 트릭을 부리게 함으로써 가능해집니다. WebSocket은 클라이언트측의 브라우저 앱이 임베디드 서버 애플리케이션과 직접 통신할 수 있도록 해 주는 새로운 기술(2012년부터 이용 가능했지만)로서 임베디드 서버 애플리케이션의 성능을 높이고, 지연시간을 줄이며, 보안을 향상시키고, 더욱 명확하게 구현될 수 있게 해 줍니다.

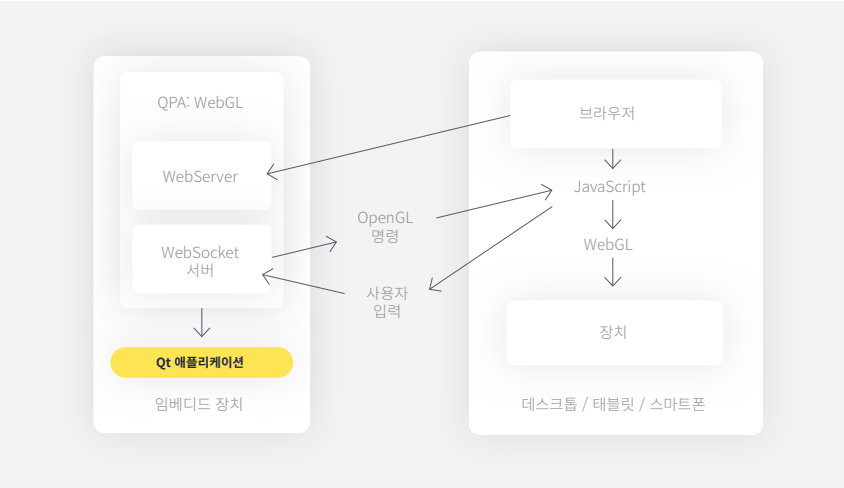
Qt를 활용한 Zero Installation 솔루션

이 백서를 읽는 분이라면 분명히 Qt 프레임워크의 몇 가지 장점에 대해 이미 알고 계실 것입니다. 이 프레임워크는 개발자, 라이브러리, 도구로 구성된 하나의 거대한 생태계로서 크로스 플랫폼 애플리케이션 구축에 뛰어나며 C++ 등의 고유한 성능을 제공합니다. Qt는 매우 유능한 C++ 프레임워크입니다. 그러면 여러분도 이것을 활용해 브라우저 기반 Zero Installation 애플리케이션을 만들 수 있을까요?

물론입니다. C++ Qt 애플리케이션을 브라우저 기반의 버전으로 변환하는 방법은 기본적으로 두 가지가 있는데, 바로 WebGL과 WebAssembly입니다. 이 두 가지 기술을 조금 더 자세히 살펴보고 무엇이 필요한지 알아보겠습니다.

WebGL

Qt WebGL 솔루션은 Qt 렌더링 명령을 WebGL 스트림으로 변환합니다. 이것이 멀리 떨어진 브라우저로 전송되어 임베디드 장치의 화면을 구성합니다. 개발자의 입장에서 이를 구현하는 것은 아주 쉬운 일입니다. 실제로 대부분의 경우 이를 완수하는 데 필요한 작업은 실행인수 (-platform webgl)를 하나 더 추가하는 것뿐입니다



Qt WebGL 블록 도표

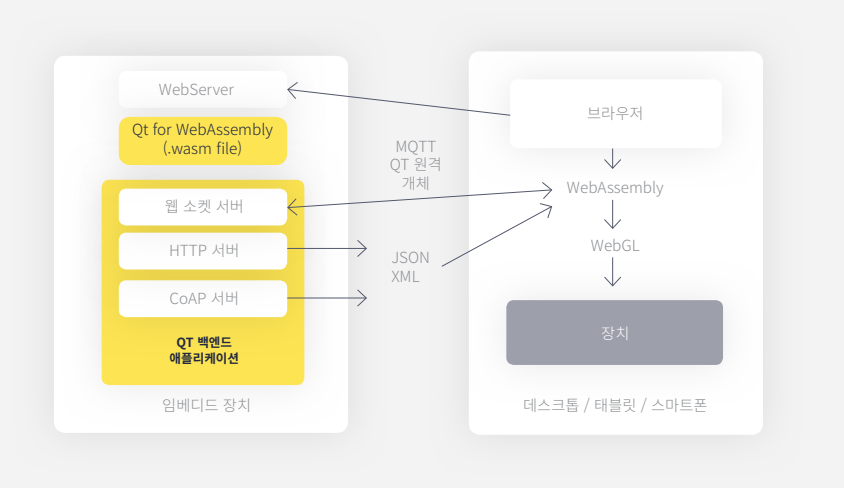
다이어그램에서 보실 수 있듯이 이 간단한 명령을 지원하기 위해 꽤 많은 일이 보이지 않는 곳에서 일어나고 있습니다. 처음에 임베디드 장치가 Zero Installation 애플리케이션용 경량 웹 서버를 작동시키고 WebSocket 서버가 명령을 송수신합니다. 브라우저가 웹 서버에 연결되면 서버는 WebSocket 연결을 생성하는 작은 JavaScript 프런트엔드 애플리케이션을 명령을 송수신하는 임베디드 장치의 WebSocket 서버로 다시 보냅니다.

그 동안 임베디드 장치는 Qt 애플리케이션을 실행하지만 QPA 플러그인이 UI에서 생성된 모든 OpenGL ES 드로우(draw) 명령을 가로챕니다. 모든 드로우 명령은 직렬화된 WebGL 스트림으로 변환되어 WebSocket 연결을 통해 JavaScript 앱으로 전송됩니다. JavaScript 앱은 드로우 명령을 포착하고 WebGL API를 사용해 브라우저 안에 임베디드 장치의 UI를 그립니다. JavaScript 앱은 이와 동일한 WebSocket 채널을 사용하여 사용자의 마우스 및 키보드 명령을 포착하고, 이를 서버에 다시 전송해 사용자 인터페이스 루프를 완성합니다.

이 솔루션에서는 서버 측이 대량의 작업을 수행합니다. 이것이 클라이언트 측 JavaScript 프로그래밍을 통해 화면을 그리는 일반적인 WebGL과 다른 점입니다. 이로 인해 여러 측면이 더욱 단순해지지만 몇 가지 한계를 드러냅니다. 다음 페이지에서 이러한 점들을 상세히 비교하였습니다.

WebAssembly

만약 Qt WebAssembly 경로로 진행하기로 했다면 WebAssembly 바이너리를 생성하기 위해서는 표준 Qt 애플리케이션이 Emscripten 컴파일러로 다시 컴파일됩니다. 이것이 클라이언트로 전달되고 브라우저 내의 컨테이너에서 실행됩니다. 클라이언트의 브라우저가 이러한 앱을 실행할 수 있는 성능을 제공하기 때문에 이 접근법으로 동시에 여러 클라이언트를 지원하기가 훨씬 더 쉬워졌습니다.



Qt WebAssembly
블록 다이어그램

WebAssembly 솔루션에 대한 이 블록 다이어그램은 구조적으로 WebGL과 유사합니다. WebAssembly를 이용하려면 클라이언트와의 연결을 처리할 수 있는 서버는 물론 애플리케이션 바이너리를 제공하는 경량 웹 서버도 필요합니다.

Qt WebGL 솔루션과 달리 프로그래머는 애플리케이션과 임베디드 장치 간의 상호작용을 특별히 관리해야 합니다. 이를 해결하기 위한 한 가지 방법은 클라이언트 앱이 실행해야 하는 각각의 특정 작업을 다루는 세분화된 모듈식(fine-grained modular) 서비스를 제공하는 마이크로 서비스 아키텍처를 사용하는 것입니다. 클라이언트의 브라우저 애플리케이션은 모든 요청을 패키징하여 임베디드 장치로 보냅니다. 장치의 서버는 요청 패킷을 수신하여 풀어서 실행합니다. (MQTT 또는 JSON은 사용하기 편리하게 하는 상응하는 Qt 클래스를 가진 두 가지 경량화된 포맷입니다.) 또 다른 대안은 QtRemoteObjects로, 이를 통해 Qt 슬롯/신호 매커니즘이 기기들 사이에서 작동할 수 있게 됩니다.

WebGL과 WebAssembly 비교

	WebGL	WebAssembly
How it works	장치가 직렬화된 WebGL 데이터 스트림을 클라이언트 브라우저로 보내 원격 UI를 표시	장치가 WebAssembly 바이너리를 클라이언트 브라우저로 전달; 앱이 브라우저 내에서 실행됨
기존 Qt 앱 변환	매우 간단함. 클라이언트 측 애플리케이션 그래픽이 자동으로 처리되며 직접 장치 제어가 영향을 받지 않음	간단함. WebAssembly 툴체인을 사용한 리컴파일로 변환이 대부분 처리됨. 하드웨어에 직접 액세스하는 구성요소는 마이크로 서비스 아키텍쳐(또는 이와 동등한 것)를 사용하기 위해 변환되어야 함
애플리케이션 속도	임베디드 하드웨어가 결정	클라이언트 장치 하드웨어가 결정
UI 응답성	UI가 임베디드 하드웨어에서 실행되고 클라이언트 장치로 전송됨으로 응답성은 충분한 네트워크 대역폭과 낮은 패킷 지연시간에 따라 결정됨	UI가 클라이언트 장치에서 직접 실행됨으로 응답성은 클라이언트 하드웨어 사양과 백그라운드에서 실행되는 다른 클라이언트 애플리케이션에 따라 결정됨
최초 시작 시간	빠름 - 거의 즉시 실행됨. 첫 번째 프레임을 표시하기 전에 클라이언트 브라우저가 작은 스텁을 받기만 하면 됨	상대적으로 느림 - 몇 초 정도 걸릴 수 있음. 전체 WebAssembly 애플리케이션이 클라이언트로 전송되어야 하며, 첫 실행 시 브라우저에서 컴파일해야 함
동시 클라이언트 수	1개. 멀티 프로세스 Qt Web-GL이 로드맵에서 진행 중임	제한 없음
리버스 엔지니어링 난이도	클라이언트 장치는 WebGL 데이터 스트림만 수신하므로 클라이언트 측 리버스 엔지니어링은 불가능	동등한 HTML5/JavaScript 제품을 사용했을 경우보다 훨씬 힘들지만, 클라이언트 장치로 전송되는 WebAssembly 코드의 리버스 엔지니어링이 가능.
한계	앱이 OpenGL 외의 드로잉을 사용 시 적용 불가능(예: Qt Widgets)	앱에 소스가 없는 써드파티 모듈이 포함된 경우 모든 코드를 WebAssembly로 재컴파일해야 하므로 적용 불가능

Qt의 축적된 Zero Installation 기술

그러면 Qt로 Zero Installation 솔루션을 구축하려면 무엇이 필요할까요?

먼저 디자이너들이 그래픽 자산을 만들기 위해서는 Adobe Photoshop 또는 Adobe Sketch가 필요합니다. 또, 여러분이 이용하는 애플리케이션의 다양한 화면에 구현할 그래픽 자산에서 선언적인 QML을 생성하는 데 사용되는 Qt Design Studio도 필요합니다.

다음으로 Qt 애플리케이션을 구축하는 데 필요한 컴파일러, 도구, 라이브러리를 제공하는 Qt Creator가 필요합니다. 이러한 통합 개발 환경(IDE)을 활용하면 QML이나 C++을 사용한 디자이너의 그래픽 자산으로 Qt 애플리케이션을 구축할 수 있습니다. (참고: Qt for WebAssembly에서 Qt 소스를 다운로드할 수 있는 옵션을 선택해야 WebAssembly를 위한 재컴파일이 가능합니다.)

또한 UI 구축(QML, Widgets, OpenGL, SVG, Qt 3D), 하드웨어 통신 관리(네트워킹, WebSocket, Bluetooth, 직렬 포트, CAN, Modbus, 센서), 데이터 관리(SQL, XML, 이미지 형식)를 위한 여러 가지 풍부한 구성요소와 도구를 제공하는 Qt for Application Development도 필요합니다.

그 다음은 Qt for Device Creation입니다. 이 패키지는 구축, 디버깅 및 임베디드 장치 배포에 필요한 필수 구성요소를 제공합니다. 여기에는 크로스 컴파일 툴 체인, USB를 통한 디버깅 소프트웨어, Boot to Qt 소프트웨어 스택, Qt for RTOS 및 다수의 레퍼런스 타깃 플랫폼이 포함되어 있습니다. 또한 Qt WebGL와 Qt for Web Assembly의 구성요소가 모두 포함되어 있기 때문에 개발자가 두 유형의 솔루션을 모두 만들 수 있습니다.

마지막으로 모든 Qt Zero Installation 솔루션을 위한 핵심 요소 중 하나로 Qt for Automation을 들 수 있습니다. Qt for Automation에는 Qt MQTT, Qt OPC UA, Qt KNX, Qt CoAP, Qt Remote Objects 등을 비롯한 임베디드 장치와 클라이언트 앱 간의 통신을 위한 키가 다수 포함되어 있기 때문입니다.

Zero Installation과 클라우드

임베디드 장치에 Zero Installation 애플리케이션을 호스팅하는 또 다른 옵션은 클라우드에 호스팅하는 것입니다. Qt는 모든 도구를 동일하게 유지한 상태에서 여러분의 개발 전략에 이 접근방법을 포함시킬 수 있는 유연성을 제공합니다. 임베디드 하드웨어가 복잡하고 많은 데이터를 요하는 앱을 호스팅하거나, 실행할 성능을 갖추지 못했거나, 앱이 클라우드에서 유지되는 데이터 리소스에 액세스해야 한다면 클라우드 기반 앱은 좋은 접근방식입니다.

클라우드 호스팅 옵션의 단점은 연결입니다. 임베디드 장치가 클라이언트 앱을 호스팅할 때 클라이언트 브라우저와 장치 간의 연결이 보장됩니다. 클라우드 기반 아키텍쳐는 클라이언트 장치를 프록시, VPN 또는 방화벽의 개입 없이 인터넷에 연결하도록 요구하며, 클라이언트 브라우저부터 임베디드 장치까지 별도의 연결을 필요로 합니다. 이러한 요구사항을 해결하는 것이 불가능하지는 않지만 "마찰 없는" 사용자 경험을 만들어 내는 것을 다소 복잡하게 만듭니다.

클라우드의 Zero Installation이 갖는 큰 장점의 하나는 네트워크 전반의 많은 장치에서 입력되는 정보를 단일 모델에 통합할 수 있다는 것으로, 이는 디지털 트윈 개발에서 매우 중요합니다. 디지털 트윈은 현실 세계의 시스템 또는 개체에서 수신된 센서 정보를 통해 실시간 가상 모형을 만듭니다. 이 가상 모형을 통해 사전 유지관리, 운영 효율성 및 교육 정보를 위해 개체를 원격으로 검사할 수 있습니다. 디지털 트윈 기술은 제조, 의료, 자동차, 건축 자동화, 우주, 국방 분야에서 큰 역할을 하고 있습니다.

기능적 안전성과 사이버 보안

사이버 보안에 대한 우려가 있을 때 Zero Installation은 이를 어떻게 해결해 나갈까요? 많은 IT 기업은 방화벽과 사용자 권한을 활용하고 있는데, 애플리케이션 소프트웨어를 설치하기 위해서는 이들을 느슨하게 해야 합니다. 브라우저에서 Zero Installation 앱을 실행하면 엄격한 IT 통제를 유지함과 동시에 사용자가 제품의 컴패니언 앱에 액세스할 수 있습니다. 브라우저는 클라이언트 측에서 일어나는 해킹 시도로부터 클라이언트 컴퓨터를 고립시켜 가상화되고 격리된 환경을 제공합니다. 하드웨어와 클라이언트 간의 연결에는 두 컴퓨터 사이를 오가는 데이터를 암호화 및 인증하기 위한 보안 소켓이 사용됩니다.

Zero Installation은 시작하기 쉽고, 유지관리 부담이 적고, 크로스 플랫폼 및 이동성 관련 기능을 제공하는 등의 여러 가지 장점을 사용자에게 제공합니다.

Zero Installation은 리버스 엔지니어링을 방지하는 데에도 도움이 됩니다. WebGL 구현에서는 애플리케이션 데이터가 전송되지 않고 WebGL 렌더링 명령만 전송되므로 클라이언트 측에서 리버스 엔지니어링이 불가능합니다. WebAssembly 애플리케이션은 클라이언트에 실행 파일을 전송하기는 합니다. 그러나 최종적인 바이너리 코드는 사람이 읽을 수 있는 동등한 수준의 HTML5/JavaScript 앱보다 훨씬 더 이해하기 어렵습니다.

Qt는 Qt Safe Renderer를 사용한 산업(IEC 61508), 자동차(ISO 26262), 의료(IEC 62304) 인증 시스템을 만드는 데 사용될 수 있습니다. 그러나 WebAssembly를 사용한 Qt Zero Installation은 임베디드 시스템에서 어떠한 렌더링도 필요로 하지 않습니다. 인증을 더 간편하게 하는 것은 바로 헤드리스(headless) 솔루션입니다. (안타깝지만 인증이 쉽지 않은 다른 WebGL 제품의 경우에도 동일하게 간편하다고 말하기는 어렵습니다.)

결론

Zero Installation은 시작하기 쉽고, 유지관리 부담이 적고, 크로스 플랫폼 및 이동성 관련 기능을 제공하는 등 여러 가지 이점을 사용자에게 제공합니다. 제품 개발자들은 종종 이러한 솔루션을 만들기 위해 Qt를 선택하고 있습니다. Qt는 크로스 플랫폼 유연성, 안정적이고 성능 기준에 부합하는 도구, 넓게 지원되는 개발자 생태계를 통해 우월한 사용자 경험을 제공하는 동시에 개발 프로세스를 단축하고 간소화할 수 있기 때문입니다.

또 다른 이점은 Qt for WebGL나 Qt for WebAssembly를 활용해 기존 네이티브 Qt 애플리케이션을 브라우저 기반 Zero Installation 앱에서 재사용할 수 있다는 것입니다. 구현이 용이하고 단일 코드 기반이라는 점 외에도 Qt 기반의 Zero Installation 애플리케이션을 이용하면 보안 및 성능 향상을 비롯한 추가적인 이점도 얻을 수 있습니다.

Zero Installation이 여러분에게 적합한 솔루션이라고 생각되신다면 Qt에 문의하십시오. 여러분이 앞으로 만드실 제품에 Zero Installation을 어떻게 구현할 수 있을지 이해하실 수 있도록 도와드리겠습니다.