**Growing a Language**

언어를 디자인할 때는 어떤 방향을 지향해야 하는가? 사용자가 요구하는 것을 모두 built-in 으로 구현한 친절한 언어? 처음부터 많은 기능을 담을 수 있는 큰 규모로 설계되는 언어? 이 강의에서는 기본적인, 많지 않은 기능을 담은 작은 규모의 언어에서 시작하여 사용자들이 그 언어를 사용하며 언어의 성장과 발전에 기여하는 방식이 바람직하다는 점, 그리고 그러안 언어를 설계할 때 주의해야 할 점에 대해서 배울 수 있었다. 처음부터 큰 규모로 언어를 디자인하는 것은 현실적으로 꽤나 힘든 일이다. 또한, 사용자들의 요구를 모두 받아들여 언어 구현에 반영하면 그러한 요구사항에 대해 동의하지 않는 사용자가 항상 있을 수 있으므로 불필요한 작업을 증가시키게 된다. 따라서 작은 규모에서 시작하는 대신 사용자들에게 그들의 요구사항을 스스로 라이브러리 등으로 구현할 수 있는 툴을 제공하여 사용자가 직접 언어의 힘을 다채롭게 만들어 나갈 수 있게 한다는 것이다.

그러면 이러한 방식을 채택함에 있어서 어떤 점을 주의해야할까? ‘성장하는 언어’ 모델에서 가장 중요한 점 중 하나는 그때그때마다 유용하다고 판단되는 기능 혹은 코드 블록을 채택하고 배포하면서 유연하면서도 최선의 선택을 찾아 발전한다는 것이다. 즉, 언어를 설계할 때 그 언어의 발전 방향을 하나로 지정해버리면 언어가 점점 큰 규모로 발전한다해도 그 언어는 더욱더 강력하게 발전할 수 있는 기회를 배제한 채 바뀌지 않는 미래를 향해 나아갈 뿐이다.

또한, 사용자가 언어를 성장시킬 수 있게 하려면 외부인이 쉽게 접근하여 영향을 끼칠 수 없는 폐쇄적인 언어이어서는 안된다. 아무리 그 당시 기준에서 훌륭한 언어라도 그 언어의 디자이너들만이 언어를 수정하고 발전시킬 수 있다면, 혹은 언어 자체가 더 나은 기능을 받아들일 수 없는 구조로 설계되었다면 그 언어의 발전 속도는 느리거나 혹은 아예 침체 상태일 것이고, 그러면 향후 10여년 이상이 흐른 뒤에는 자연스레 다른 발전하는 언어에 의해 밀릴 수 밖에 없을 것이다. 제아무리 당대 최고의 제품이라 할 지라도 바뀌는 것이 없으면 세월이 지난 뒤에는 구식이 되고 만다. 언어 역시 마찬가지이다. 발전하는 언어가 되기 위해서는 많은 사람들이 개발과 보완 및 피드백 과정에 참여할 수 있는 개방적인 환경이 마련되어야 한다.

과거에는 프로그래밍 언어의 수가 그렇게 많지 않았다. 그럼에도 불구하고, 이 강의에 따르면 그 당시에도 사용자들의 도움에 힘입어 성공적으로 발전한 언어와 발전하지 못해 결국 실패하게 된 언어가 있었다. 현재에는 더욱더 많은 프로그래밍 언어가 존재한다. 그러므로 다른 언어보다 뛰어난 새로운 멋진 언어를 설계하고자 한다면, 혹은 기존에 존재하는 언어를 보다 더 발전시키고자 한다면, 사용자들이 언어를 사용하며 손쉽게 새로운 기능을 구현한 코드 블록을 만들 수 있도록 개방적이고 미래 지향적인 언어 패턴을 고안해야 한다는 점을 반드시 기억해야 한다는 것이 이 강의의 교훈이다.