ようこそテキストプログラミングへ

ようこそ、テキストプログラミングの世界へ！

テキストプログラミンでは、Scratch のようなプログラムをパズルのように組み立てていくのではなく、テキスト(文字)によってプログラムを組み立て行きます。なんとなく難しそうですが、プログラムの本質は Scratch と同じなため、Scratch を理解できていれば敷居はそれほど高くはありません。それでは、頑張ってテキストプログラミングをしていきましょう！

プログラミング言語

テキストプログラミングと言っても、世の中には様々な言語があります。有名なところで言えば、C 言語、Java、C#、JavaScript、PHP、Ruby・・・・など多種多様です。それぞれ、目的によって使い分けます。電子工作に組み込むプログラムには C 言語、Android のアプリは Java、Windows のアプリは C#、Web で使われるのは PHP や Ruby・・・といった感じにそれぞれの言語でよく使われている場所が異なります。その中で Coder Dojo では JavaScript(略して JS)を扱っていきます。

JavaScript について

JavaScript はホームページを動かすために作られた言語です。例えば、ボタンを押したらメニューが出てくるようなホームページや、ポップアップが表示されるようなページ、スマホで画像をスライドすると次の画像に切り替わったりするホームページ等々そのようなプログラムは大体 JavaScript という言語でプログラミングされています。一般的ではないですが JavaScript でもゲームを作ることは可能です。そのやり方を Coder Dojo で学んでいきます。

どうして JavaScript なのか

JavaScript はゲームを作るためには一般的な言語ではありません。ですが、手軽なので今回はこの言語を採用させていただきました。JavaScript で書かれたプログラムはブラウザ(Internet Explorer,Safari, Google Chrome 等)がすぐに動かすことができますし、Mac だろうと Windows だろうと、同じ手順で作って動かすことができます。ほかの言語ですと、動かすためにいろいろとソフトウェアをインストールしないといけないのが大変です。また OS によってやり方、プログラムの書き方、動きが異なることがあるので、様々な PC を持ってきている Dojo の環境には合致しませんでした

Javascript だけではゲームは作りにくい！？

先ほど述べたように Javascript とは Web ページを動かすために作られた言語ですので、ゲームを作るにはちょっとやりにくい側面があります。そこで**ライブラリ**というプログラミング言語を補助してくれるものを使います。料理に例えると、スパイスをいろいろブレンドしてカレーを作り上げるのは結構手間がかかる作業です。そこで、カレールーを使えばカレーはより簡単につくることができます。また、もっと簡単に作ろうとすればレトルトカレーを買ってこればよいでしょう。このような料理でいうカレールーやレトルトカレーのような存在である、手間のかかるところをやってくれいて、あとは簡単なことの組み合わせでできるようにしてくれているものをライブラリと呼ぶのです。そのライブラリに **enchant.js** というものを利用しました。これは JavaScript でゲームを作る人が比較的簡単に作れるように作られたライブラリですので、よりゲームを簡単に Javascript で作成できるようになります。

なので、困ったことがあれば「enchant.js」で調べてみると有益な情報が出てくるかもしれません。

プログラミングの歴史

ちょとだけプログラミング言語の歴史にも触れておきます。
大昔の 1800 年～1900 年ごろに初めてプログラミングの前身となるものが出来上がってきました。
それまでのコンピュータは計算の式や計算式の中に入っている数字を変えたい！と思ったらコンピュータの回路を変える必要がありました。1800 年ごろに作られた織機(布を作る機械)では穴の開いた紙を織機に入力すると、穴の開いている位置によっていろいろなパターンの布をつくことができました。これがプログラミングの起源と言われていました。ちなみに、コンピュータの動作を変えるための穴を開ける紙をパンチカードと呼びます。
1940 年ごろになると、プログラムの実行順序をメモリに入れておくことができるようになりました。この計算をしたら、次はこの計算をして・・・という順序が、今までは機械仕掛けだったのが、読み書きが簡単にできるメモリに入力されることでかなりプログラムの自由度が上がります。例えば足し算なら、引き算なら「10」、掛け算なら「11」などのように、数字でどのような計算をするかを順番に書いていきました。これのことを**機械語**と呼びます。
ただ、これではとても読みにくいですね。
なので、それぞれの処理を英語で書いて、あとで機械語に変えるという手法がとられました。例えば 1+2 であれば「ADD 1 2」のように記述するようになっていき、それを機械語に変更してから実行されるようになりました。この機械語を置き換えた言語のことを**アセンブリ言語**と呼びます。

このアセンブリ言語にも問題がありました。一つはコンピュータが違うと言語が違うので、コンピュータを変えるたびに新しい言語を覚えなければならないこと、もう一つはちょっとでも複雑なことを書こうとするとたくさんのことを書かないといけなくなってしまい、すぐに読みにくくなってしまうことです。例えば「2+3×3+4 ÷ 5」のような計算式はアセンブリ言語で書くのはちょっと大変です。そこで登場したのが**プログラミング言語**です。各コンピュータの違いを意識しなくてもかけるようになっており、また、「a = 2 + 3 * 3 / 4」のように先ほどの計算も一行で書くことが可能になります。このプログラミング言語は実行する前にアセンブリ言語に変換され、最終的には機械語に変換されます。このプログラミング言語が登場したのが 1954 年ごろの話です。このころに作られた COBOL、FORTRAN といった言語はいまだに一部では使われています。

1960 年代後半になると、コンピュータ部品の低価格化などの影響でキーボードでプログラミング言語を打ち込むということが始まりました。今の形にかなり近くなります。

また、コンピュータの性能も著しく向上してきます。プログラミング言語を機械語に直す作業のことを**コンパイル**と呼ぶのですが、これは実行前にあらかじめ行っておくのが普通でした。しかしコンピュータの性能向上のおかげで実行しながらプログラミング言語を解読していく余裕ができたため、事前にコンパイルをしなくとも実行中に適宜プログラミング言語を解読していきながら実行するような言語が出現し、普及してきました。このような形の言語のことを**インタプリタ言語**と呼びます。今回学習する JavaScript もインタプリタ言語になるので、皆さんにコンパイルをしていただく必要はありません。