「鉄道模型シミュレーターNX」で学ぶ

Pythonプログラミング入門

（パイソン）

［セレクト版］

第3章

「Python」制御と自動運転の基本

本章では、「車両」や「ポイント」などの「オブジェクト制御方法」と、「センサー」を使った「列車の出発・停車」や「往復運転」など、基本的な「自動運転スクリプト」を紹介します。

電気機関車たち。左から「EF510」「EF66」「EF81」

3-1 列車の操作

「VRM-NX」の列車は、線路の上を走らせるだけでなく、多種多様なギミックがあります。これらの多くは、「Python」で制御できます。

「編成」と「車両」

車両ごとのギミックを操作するには、「編成オブジェクト」に複数内包される「車両オブジェクト」を指定します。

(A)「「GetCar(n)関数」で、「n両目」の車両を直接指定する」か、(B)「「GetCarList()関数」で、全車両から「条件式」などで特定の車両を指定」します。

編成と車両を定義するスクリプト

```
# ID[1]の編成オブジェクトを取得
train = vrmapi.LAYOUT().GetTrain(1)

# 3両目の車両を直接指定(先頭は0)
car = train.GetCar(2)
car.SetRoomlight(True)

# 全ての中間車で室内灯をON
for car in train.GetCarList()
    if car.GetCarPos() == 0:
        car.SetRoomlight(True)
```

■「ヘッドライト」と「テールライト」

「先頭車」や「機関車」にある「ヘッドライト」(前照灯)と「テールライト」(尾灯)は、それぞれ「SetHeadlight(bool)」「SetTaillight(bool)」で「点灯/消灯」を操作できます。

それぞれの命令は、「列車の向き」に合わせて点灯します。

「ヘッドライト」と「テールライト」を同時に点灯させる、いわゆる「フル点灯」をさせる場合は、向きに関係なく点灯する「SetNDHeadlight(int, bool)関数」「SetNDTaillight(int, bool)関数」を使います。

「ヘッドライト」と「テールライト」の制御スクリプト

```
# ヘッドライト点灯（Falseで消灯）
car.SetHeadlight(True)
# テールライト点灯（Falseで消灯）
car.SetTaillight(True)

# マイナス側のヘッドライト点灯（Falseで消灯）
car.SetNDHeadlight(0, True)
# プラス側のヘッドライト点灯（Falseで消灯）
car.SetNDHeadlight(1, True)

# マイナス側のテールライト点灯（Falseで消灯）
car.SetNDTaillight(0, True)
# プラス側のテールライト点灯（Falseで消灯）
car.SetNDTaillight(1, True)
```

■ 入換標識灯

「入換標識灯」は、「貨物」などの車両を「入換」している「機関車」が、向かって左側の赤色灯を一灯点灯して掲出するものです。

「SetSCIndicator(bool)関数」で、「入換標識灯」の表示を操作できます。

入換標識灯の制御スクリプト

```
# 入換標識灯（Falseで消灯）
car.SetSCIndicator(True)
```

左から「消灯」「ヘッドライト」「テールライト」「入換標識灯」

■ 方向幕、LED

列車には、「路線名」や「行き先」を表示する、「方向幕」があります。

これらは「SetRollsignLight(bool)」や「SetLEDLight(bool)」で、「発光」を操作します。

一部の特急にある「LEDヘッドマーク」も、「SetLEDLight(bool)」で点灯します。

「方向幕」と「LED」の制御スクリプト

```
# 方向幕点灯（Falseで消灯）
car.SetRollsignLight(True)
# LED点灯（Falseで消灯）
car.SetLEDLight(True)
```

「LEDヘッドマーク」の点灯

■ 運転台・室内灯

車内の「室内灯」は、「SetRoomlight(bool)」で「点灯/消灯」を操作できます。

「SetCabLight(bool)関数」は、新しい車両規格で利用できるもので、「客室」と「運転台」の「室内灯」を、個別に操作できます。

「運転台」と「室内灯」の制御スクリプト

```
# 室内灯点灯（Falseで消灯）
car.SetRoomlight(True)
# 運転台室内灯点灯（Falseで消灯）
car.SetCabLight(True)
```

室内灯の「OFF」（上）/「ON」（下）

■ 発煙

「蒸気機関車モデル」は、「煙突」と「シリンダー」から、「黒煙」と「蒸気」をそれぞれ発生させます。

「発煙」は、「SetSmoke(bool)関数」で操作します。

発煙の制御スクリプト

```
# 発煙（Falseで消煙）
car.SetSmoke(True)
```

「煙OFF」（左）と「煙ON」（右）

■ 「パンタグラフ」の昇降

「パンタグラフ」は、電車が「架線」から電気を得るための装置です。

「SetPantograph(int, bool)」で「パンタグラフ」の「上昇/下降」を操作できます。

1つ目の引数で、「パンタグラフ番号」を指定します。

「VRM-NX」では、1車両に4つまでのパンタグラフを定義可能で、車両の「パンタグラフ定義数」は、「GetCountOfPantograph()関数」で取得可能です。

＊

一般的な電車や電気機関車の「パンタグラフ」は、1車両あたり0〜2個装備されています。

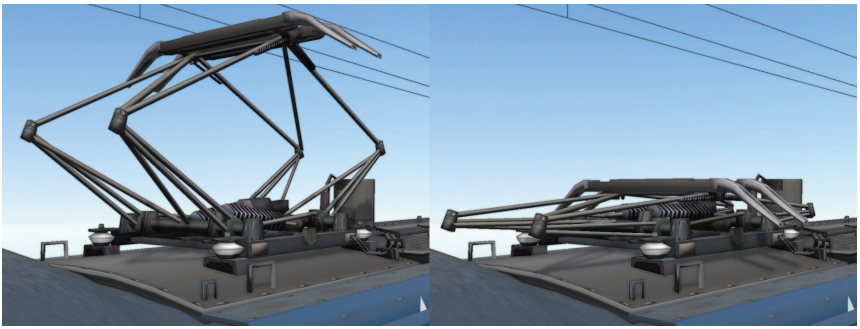
昔の電車は、「電動車」と「付随車」の「ユニット方式」が基本だったため、2両に1両の割合で「パンタグラフ搭載車両」が使われていました。

比較的新しい電車は、「引通線」を通して取り込んだ電気を編成全体で共有するため、使われる「パンタグラフ」は編成につき1〜2個が一般的です。

パンタグラフの制御スクリプト

```
# 車両のパンタグラフ数を取得(0〜4)
n = car.GetCountOfPantograph()

# パンタグラフ1を上昇(Falseで下降)
car.SetPantograph(0, True)
# パンタグラフ2を上昇(Falseで下降)
car.SetPantograph(1, True)
```



「パンタグラフ」の上昇（左）と下降（右）

■ ヘッドマーク

機関車などには、愛称などを掲げる「表示幕」(ヘッドマーク)を「SetHeadmarkDisp(int, bool)関数」で付けることができます。

引数の一つで「ヘッドマーク番号」(ヘッドマークの表示向き)を指定します。

車両にいくつのヘッドマークがあるかは、「GetCountOfHeadmark()関数」で取得可能です。

ヘッドマークの表示スクリプト

```
# 車両のヘッドマーク数を取得(0〜4)
n = car.GetCountOfHeadmark()

# ヘッドマーク1を表示(Falseで非表示)
car.SetHeadmarkDisp(0, True)
# ヘッドマーク2を表示(Falseで非表示)
car.SetHeadmarkDisp(1, True)
```

機関車の「ヘッドマーク」表示例

「ヘッドマーク」は、組み込みテクスチャーを自作して登録することもできます。

EF-66の製品組み込みテクスチャー

編成を選択した状態で、「レイアウターツール」→「ツールオプション」→「プロパティ」→「車両TEX/ヘッドマーク」からリソースを選択し、「ユーザー車両テクスチャー」を選択します。

リソースを選択する

「SetTexture(int)」で、Pythonからも操作できます。

```
# 車輌テクスチャーのリソースID:0を有効化
car.SetTexture(0)
```

■ オプション表示

「オプションパーツ」をもつ車両は、「SetOptionDisp(int, bool)」で表示が切り替えられます。

「オプションパーツ」は、主に先頭車の「連結器」や「幌表示」に割り当てられています。

オプション表示のスクリプト

```
# オプション1を有効（Falseで無効）
car.SetOptionDisp(0, True)
# オプション2を無効（Trueで有効）
car.SetOptionDisp(1, False)
```

新幹線連結器の開閉

「VRM-NX」は、編成を別の編成にゆっくり（最高速度の25％以下）近付けることで、自動で連結します。

反対に、「SplitTrain関数」を使うことで、切り離しも可能です。
連結器の表示切り替えと合わせて、「併結運転」を楽しむことができます。

新幹線併結時

在来線車両では、「貫通扉の開閉」や「貫通幌の表示」が可能です。

在来線車両の「オプション表示」切り替え

■ ドア

「NX規格」の車両では、「ドアの開閉」が「モーション」で実装されました。

「OpenDoor(int)」または「OpenDoor_Side(int, bool)」で指定したドアが開き、「車側灯」が点灯します。

「CloseDoor(int)」または「CloseDoor_Side(int, bool)」でドアが閉じ、「車側灯」が消灯します。

「駅ストラクチャー」の「ホームドア」と組み合わせることで、リアルな駅発着シーンを表現できます。

ドアの制御スクリプト

```
# 扉1を開く(扉番号は0開始)
car.OpenDoor(0)
# 扉1を閉じる(扉番号は0開始)
car.CloseDoor(0)

# 左側客用扉を開く(0:左、1:右、False:客室扉、True:業務扉)
car.OpenDoor_Side(0, False)
# 左側客用扉を閉じる(0:左、1:右、False:客室扉、True:業務扉)
car.CloseDoor_Side(0, False)
```

■著者略歴
角　卓（すみ・まさる）

大阪府四條畷市出身。
大阪工業大学情報システム学科卒業後、システムインテグレーター企業に入社。
プログラミングの他にデータベースおよびアーキテクチャの設計、インフラ構築やプロジェクト推進など、システム開発の幅広い領域で活躍中。
月刊 I/O の連載や、『鉄道模型シミュレーター NX 入門』（共著）を執筆。

［協　力］

▼ばんえつライン@ Twitter
https://twitter.com/Ban_etsu_L_VRM
▼うつみ@ Twitter
https://twitter.com/utumi787
▼USO800鉄道@ Twitter
https://twitter.com/uso800railway
▼ zio@Twitter
https://twitter.com/ziorossolabo
▼（株）アイマジック
http://www.imagic.co.ljp/
▼鉄道模型シミュレータークラウド
https://vrmcloud.net/

質問に関して

本書の内容に関するご質問は、

①返信用の切手を同封した手紙
②往復はがき
③ FAX(03)5269-6031
（ご自宅の FAX 番号を明記してください）
④ E-mail　editors@kohgakusha.co.jp

のいずれかで、工学社編集部宛にお願いします。電話によるお問い合わせはご遠慮ください。

●サポートページは下記にあります。
【工学社サイト】http://www.kohgakusha.co.jp/

I/O BOOKS
「鉄道模型シミュレーター NX」で学ぶ Python プログラミング入門【セレクト版】

2021 年 5 月 18 日　初版発行　ⓒ 2021

著　者　角　卓
発行人　星　正明
発行所　株式会社工学社
〒 160-0004
東京都新宿区四谷 4-28-20 2F
電話　(03)5269-2041(代)［営業］
(03)5269-6041(代)［編集］
振替口座　00150-6-22510

※この冊子は非売品です。