Train Sim World 5 — Unofficial API Documentation (First Release)

(Generated from ThirdRails TSWDataService usage)

Introduction

```
ThirdRails was one of the first apps to use the TSW API.
Based on the functions used in the ThirdRails TSWDataService class this documentation is generated.

Base URL: http://localhost:31270/

Enable: start TSW with launch parameter: -HTTPAPI

API Key file: Documents\My Games\TrainSimWorld5\Saved\Config\CommAPIKey.txt
Header required on every request: DTGCommKey: <your_api_key>
```

Root API Endpoints

GET /info


```
{
  "Meta": {
    "Worker": "DTGCommWorkerRC",
    "GameName": "Train Sim World 5®",
    "GameBuildNumber": 698,
    "APIVersion": 1,
    "GameInstanceID": "AB3B31BF4B7A9D571ECC2AB0143FBDB1"
  },
  "HttpRoutes": [
    { "Verb": "GET", "Path": "/info", "Description": "Get information about available commands." },
    { "Verb": "GET", "Path": "/list", "Description": "List all valid paths for commands." },
    { "Verb": "GET", "Path": "/get", "Description": "Get a value for a node's endpoint." },
    { "Verb": "PATCH", "Path": "/set", "Description": "Set a value on certain, writable endpoints."
    { "Verb": "GET", "Path": "/subscription", "Description": "Read a subscription set." },
    { "Verb": "POST", "Path": "/subscription", "Description": "Create a subscription." },
    { "Verb": "DELETE", "Path": "/subscription", "Description": "Unsubscribe from a path." },
    { "Verb": "GET", "Path": "/listsubscriptions", "Description": "List all subscriptions currently
  ]
}
```


GET /list (example shown later under Current Drivable Actor)

GET /get/{path}
`Example: GET /get/CurrentDrivableActor.Function.HUD_GetSpeed`

PATCH /set/{path}
`Example: PATCH /set/CurrentDrivableActor/Pantograph_F.Value?Value=true`

Subscriptions: /subscription (GET/POST/DELETE), /listsubscriptions

TSWDataService — Public Methods and Endpoints

IsApiAvailableAsync()

```
C#
public async Task<bool> IsApiAvailableAsync()
HTTP
GET /info
Returns: bool (true if 200 and valid JSON; else false)
```

IsDriveableActorAvailableAsync()

```
C#
public async Task<bool> IsDriveableActorAvailableAsync()
Logic
- Check drivable actor availability via internal probe
- OR fall back: GetPlayerLocationAsync() != null
Returns: bool
```

GetSpeedAsync()

```
C#
public async Task<double> GetSpeedAsync()
HTTP
GET get/CurrentDrivableActor.Function.HUD_GetSpeed
Example Response (observed):
{
  "Speed (ms)": 23.42
}
Notes:
- Unit: meters/second
- 0 returned on error
```

GetGradientAsync()

```
C#
public async Task<float> GetGradientAsync()
HTTP
GET get/DriverAid.Data
Example Response (abridged):
{
  "Result":"Success",
  "Values":{
    "signalSeen":true,
    "speedLimitSeen":true,
    "distanceToSignal":24314.767578125,
    "distanceToNextSpeedLimit":26771.4921875,
    "gradient":0,
    "signalAspectClass":"Clear",
    "signalPropertyGuid":"A094B8AA4048542749A61095CECF9984",
    "bSignalIsPermissive":false,
    "speedLimit":{"value":13.888889312744141},
    "nextSpeedLimit":{"value":11.111111640930176},
    "currentSpeedLimitSource":"TrackSpeedLimit",
    "trackMaxSpeed":{"value":13.888889312744141},
    "serviceMaxSpeed":{"value":3.4028234663852886e+38},
    "formationMaxSpeed":{"value":55.555557250976562},
    "nextSignalPosition":{"x":-24808.0625,"y":-4111.296875,"z":42203.1875},
    "nextSpeedLimitPosition":{"x":-27253.125,"y":-4349.8125,"z":42210.55859375},
    "nextSpeedLimits":[
      {
        "value":{"value":11.111111640930176},
        "distanceToNextSpeedLimit":26766.4921875,
        "nextSpeedLimitPosition":{"x":-27253.125,"y":-4349.8125,"z":42210.55859375},
        "restrictionType":"TrackRuleSpeedLimit"
```

```
      }
      /* …more entries… */
    ],
    "nextSignals":[
      { "value":"Clear", "distanceToNextSignal":24309.767578125, "nextSignalPosition":{"x":0,"y":0,"
      /* …more entries… */
    ]
  }
}
Notes:
- Speed values in m/s
- Large floats (e.g., 3.4028e+38) often mean 'undefined'
- Arrays provide lookahead for speed limits and signals
```

GetCalculatedGradientAsync() [deprecated]

```
C#
public async Task<float> GetCalculatedGradientAsync()
HTTP
GET get/DriverAid.TrackData
Notes:
- Calculates gradient from TrackData vs last position
- Differs from HUD gradient; prefer GetGradientAsync()
```

GetScenarioLocalTimeAsync()

```
C#
public async Task<string> GetScenarioLocalTimeAsync()
HTTP
GET get/TimeOfDay.data
Example Response:
{
  "Result": "Success",
  "Values": {
    "LocalTime": 6.3483732386065267e+17,
    "LocalTimeISO8601": "2012-09-20T10:06:26.065+00:00",
    "GMTOffset": 0,
    "WorldTime": 6.3483732386065267e+17,
    "WorldTimeISO8601": "2012-09-20T10:06:26.065Z",
    "DayPercentage": 0.417430579662323,
    "SunriseTime": "+05:02:36.063",
    "SolarNoonTime": "+11:06:32.032",
    "SunsetTime": "+17:10:28.000",
    "SunPositionAzimuth": 249.78106689453125,
    "SunPositionAltitude": 40.335109710693359,
    "MoonPositionAzimuth": 207.74325561523438,
    "MoonPositionAltitude": -1.8571348190307617,
    "OriginLatitude": 48.744300842285156,
    "OriginLongitude": 11.438400268554688,
    "SystemTime": 6.3893959628784e+17,
    "SystemTimeISO8601": "2025-09-20T10:07:08.784Z"
  }
}
Notes:
- C# helper formats LocalTimeISO8601 to HH:mm:ss when returning string
```

GetPlayerLocationAsync()

```
C#
public async Task<TSWLocation> GetPlayerLocationAsync()
HTTP
GET get/DriverAid.PlayerInfo
Example Response:
{
  "Result": "Success",
  "Values": {
```

```
    "geoLocation": {
      "longitude": 12.975223245531074,
      "latitude": 47.836978870405197
    },
    "currentTile": { "x": 118, "y": 96 },
    "playerProfileName": "Beentrain",
    "cameraMode": "FirstPerson_Driving"
  }
}
```

GetLocationAsync()

```
C#
public async Task<TSWLocation> GetLocationAsync()
HTTP
GET get/CurrentDrivableActor.LatLon
Notes:
- If not available, falls back to GetPlayerLocationAsync()
- Response structure may vary by vehicle; typically provides latitude/longitude fields
```

GetWeather()

```
C#
public async Task<WeatherValues> GetWeather()
HTTP
GET get/WeatherManager.Data
Example Response:
{
  "Result": "Success",
  "Values": {
    "Temperature": 12.3,
    "TemperatureOverridden": 0,
    "Cloudiness": 0.45,
    "CloudinessOverridden": 0,
    "Precipitation": 0.0,
    "PrecipitationOverridden": 0,
    "Wetness": 0.2,
    "WetnessOverridden": 0,
    "GroundSnow": 0.0,
    "GroundSnowOverridden": 0,
    "PiledSnow": 0.0,
    "PiledSnowOverridden": 0,
    "FogDensity": 0.1,
    "FogDensityOverridden": 0
  }
}
```

GetCurrentDrivableActorAsync()

```
C#
public async Task<CurrentDrivableActor> GetCurrentDrivableActorAsync()
HTTP
LIST /list/CurrentDrivableActor
Example Response (abridged):
{
  "Result": "Success",
  "NodePath": "CurrentDrivableActor",
  "NodeName": "CurrentDrivableActor",
  "Nodes": [
    { "Name": "Throttle_F" },
    { "Name": "TrainBrake_F" },
    { "Name": "Reverser_F" },
    { "Name": "Pantograph_F" },
    { "Name": "Cab_F" },
    { "Name": "LZB_Isolation" }
    /* … many more … */
  ],
  "Endpoints": [
    { "Name": "Function.HUD_GetSpeed", "Writable": false },
    { "Name": "Function.HUD_GetAcceleration", "Writable": false },
    { "Name": "Function.HUD_GetTrainBrakeHandle", "Writable": false },
    { "Name": "ObjectClass", "Writable": false }
    /* … many more … */
  ]
}
Notes:
- Content varies by locomotive
- Use /set for writable endpoints
```

IsAWSFlowerActive() / GetAWSWarningAlert()

```
C#
public async Task<bool> IsAWSFlowerActive()
public async Task<bool> GetAWSWarningAlert()
HTTP
GET get/CurrentDrivableActor/{Node}.{Endpoint}
Example Response:
{ "Result":"Success", "Values": { "Value": true } }

Notes:
- Node & endpoint resolved dynamically based on feature key
- Returns bool
```

SetFeatureStatus(featureKey, value)

```
C#
public async Task<bool> SetFeatureStatus<T>(string featureKey, T value)
HTTP
PATCH set/CurrentDrivableActor/{Node}.{Endpoint}?Value={value}

Examples:
- Raise pantograph:
  PATCH /set/CurrentDrivableActor/Pantograph_F.Value?Value=true

- Set array-like values (sent as multiple calls):
  PATCH /set/CurrentDrivableActor/ConsoleLightsDimmer_F.Value?Value=0.2
  PATCH /set/CurrentDrivableActor/ConsoleLightsDimmer_F.Value?Value=0.5
```

GetCurrentFormationLength()

```
C#
public async Task<int> GetCurrentFormationLength()
HTTP
```

```
GET get/CurrentFormation.FormationLength
Example Response:
{ "FormationLength": 200 }

Notes:
- Method adjusts for Multiple Unit (MU) detection via formation names
- Returns -1 on error
```

GetLocoNameAsync()

```
C#
public async Task<string> GetLocoNameAsync()
Logic
- Attempts to resolve current vehicle ObjectClass via vehicle id
- Fallback: GetLocoNameAt(0)
- Returns "Unknown Loco" on failure
Example Output:
"RVM_CRG_DB_BR101_C"
```

GetLocoNameAt(index)

```
C#
private async Task<string> GetLocoNameAt(int index)
HTTP
GET get/CurrentFormation/{index}.ObjectClass
Example Response:
{ "ObjectClass": "RVM_CRG_DB_BR101_C" }
```

Reset()

```
C#
public void Reset()
Effect
Clears cached data (e.g., after loco change).
```

Appendix — Headers & Examples

```
All requests must include:
DTGCommKey: <your_api_key>

cURL Examples:
- Read speed
  curl -H "DTGCommKey: <key>" "http://localhost:31270/get/CurrentDrivableActor.Function.HUD_GetSpeed

- Raise pantograph
  curl -X PATCH -H "DTGCommKey: <key>" "http://localhost:31270/set/CurrentDrivableActor/Pantograph_F
```