Web Service API

DriverTech offers an API to allow Clients to use their own back end systems instead of or in addition to the FleetWatcher Web Portal.

This API, known as the FleetWatcher API, is comprised of secure enterprise ready Web Services hosted by servers and networks designed for high availability. The services included in the FleetWatcher API are based upon the same core types as defined in the XSD/WSDL ensuring consistency. Also included is verbose error and exception reporting to speed initial development and enhance the ability to resolve problems programmatically.

- Admin Service — The Admin Service provides administrative operations for Drivers, Forms, Reporting Groups Assignments
- Data Service — The Data Service provides communications and other information used on a daily basis by Company's TruckPC's, Drivers, Managers, etc
    - Include Destinations in an Outbound Message — Including one or more destination locations in an outbound message allows the Driver to route from their current location to the selected destination.
- Reports Service — The Reports Service provides all reporting capabilities including End of Day Rollup (EODR) raw data.

Service Release Status

Service	Version	Rev. Date	Upgrade Info
Administration	**2.0.106**	**03/24/2022**	**Release Notes**
Data	**2.1.82**	**02/28/2018**	**Release Notes**
Reports	**2.0.100**	**11/01/2017**	**Release Notes**

DEPRECATING

Data v2.0	2.0.80	03/12/2018	2.0 -> 2.1

DEPRECATED

Data v2.0.Test	2.0.80	05/21/2013	2.0ut -> 2.1
Data v1.4	1.4.50	05/20/2013	1.2,1.4 -> 2.1
PreRelease	2.0.75	05/20/2013	PR -> 2.1

Implementation Overview

1. **All DateTime values are in UTC / GMT** - both consumed and provided by the service (see XSD Date Time)
2. **Many common types are re-used throughout all services** to enable developers to apply both DRY, and to a greater extent, SOLID principals:
    a. ***DriverBase*** - provides only enough detail required to identify a specific Driver or Operator
    b. ***TruckPCBase*** - provides only enough detail required to identify a specific Device or ELD
    c. ***Position*** - provides enough detail to pinpoint a location
    d. ***FormMessage*** - provides detail of the content for a particular instance of a Macro or Form
3. **Detailed Exception Reporting** is provided via Soap Faults. **VENDORS MUST, at a minimum, catch and handle ClientFaults vs ServerFaults appropriately**:
    a. ***Server Faults*** - Should be retried after a limited number of escalating periods of time

Server Fault Example

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
   <soap:Fault>
    <faultcode>soap:Server</faultcode>
    <faultstring>Server Unable to Handle Request. SupportCode:d08bdaad-179c-4194-bd7b-6f5fd669ab93<
/faultstring>
    <faultactor>http://fwapi.drivertech.com/Data/2_1/DataService.asmx</faultactor>
    <detail>
     <fault>
      <Errors xmlns="http://fwapi.DriverTech.com/CommonTypes/2009/04">
       <Error ErrorCategoryId="100" ErrorCodeId="100">
        <ErrorCode>Server</ErrorCode>
        <ErrorCategory>Server</ErrorCategory>
        <ErrorDescription/>
       </Error>
      </Errors>
     </fault>
    </detail>
   </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

b. ***Client Faults*** - Shall not be retried as any additional attempt will invoke the same failed response UNLESS/UNTIL some external action is taken

Client Fault - Schema Validation

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
   <soap:Fault>
    <faultcode>soap:Client</faultcode>
    <faultstring>XmlSchemaValidationException: The 'http://fwapi.DriverTech.com/CommonTypes/2009/04:
CompanyCode' element is invalid - The value '?' is invalid according to its datatype 'String' - The actual
length is less than the MinLength value.</faultstring>
    <faultactor>http://fwapi.drivertech.com/Data/2_1/DataService.asmx</faultactor>
    <detail>
     <fault>
      <Errors xmlns="http://fwapi.DriverTech.com/CommonTypes/2009/04">
       <Error ErrorCategoryId="1000" ErrorCodeId="1109">
        <ErrorCode>XmlSchemaValidationFailed</ErrorCode>
        <ErrorCategory>Client</ErrorCategory>
        <ErrorDescription/>
       </Error>
      </Errors>
     </fault>
    </detail>
   </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Client Fault - Incomplete Request

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Header>
      <ResponseHeader xmlns="http://fwapi.DriverTech.com/CommonTypes/2009/04">
         <DriverTechAPIVersion>2.1.40.27252</DriverTechAPIVersion>
         <RequestId>11111</RequestId>
         <ExecutionMilliseconds>0</ExecutionMilliseconds>
      </ResponseHeader>
   </soap:Header>
   <soap:Body>
      <soap:Fault>
         <faultcode>soap:Client</faultcode>
         <faultstring>Client: Must provide at least 1 OutboundMessageRequest Item SupportCode:12403e07-83f2-
4e85-9fe5-9dc2b2c18aa2</faultstring>
         <faultactor>http://fwapi.drivertech.com/Data/2_1/DataService.asmx</faultactor>
         <detail>
            <fault>
               <Errors xmlns="http://fwapi.DriverTech.com/CommonTypes/2009/04">
                  <Error ErrorCategoryId="1000" ErrorCodeId="1000">
                     <ErrorCode>Client</ErrorCode>
                     <ErrorCategory>Client</ErrorCategory>
                     <ErrorDescription>Must provide at least 1 OutboundMessageRequest Item</ErrorDescription>
                  </Error>
               </Errors>
            </fault>
         </detail>
      </soap:Fault>
   </soap:Body>
</soap:Envelope>
```

c. ***Fault Detail***

Enumeration of ErrorCategory and ErrorCode

```
    /// <summary>
    /// Parent Category of an ErrorCode
    /// </summary>
    [Serializable]
    [XmlType(TypeName = "ErrorCategory", AnonymousType = true)]
    public enum DTErrorCategory
    {
        /// <summary>
        /// Generic Server Error
        /// </summary>
        [EnumTextValue(@"Generic Server Error")]
        Server = 100,
        /// <summary>
        /// Generic Client Error
        /// </summary>
        [EnumTextValue(@"Generic Client Error")]
        Client = 1000,
        /// <summary>
        /// Client schema error
        /// </summary>
        [EnumTextValue(@"Client schema error")]
        InvalidSchema = 1050,
        /// <summary>
        /// Client versioning error
        /// </summary>
        [EnumTextValue(@"Client versioning error")]
        VersionMismatch = 1051,
        /// <summary>
        /// Error authenticating to access webservices
        /// </summary>
        [EnumTextValue(@"Error authenticating to access webservices")]
        Authentication = 1500,
        /// <summary>
        /// Invalid data found in Request
        /// </summary>
```

```
    [EnumTextValue(@"Invalid data found in Request")]
    InvalidRequestData = 2000,
}
/// <summary>
/// Specific code of an encountered Error.
/// </summary>
[Serializable]
[XmlType(TypeName = "ErrorCode", AnonymousType = true)]
public enum DTErrorCode
{
    /// <summary>
    /// Default Error Code
    /// </summary>
    [EnumTextValue(@"Default Error Code")]
    Unavailable = 1,
    /// <summary>
    /// Generic Server Error
    /// </summary>
    [EnumTextValue(@"Generic Server Error")]
    Server = 100,
    /// <summary>
    /// Generic Client Error
    /// </summary>
    [EnumTextValue(@"Generic Client Error")]
    Client = 1000,
    /// <summary>
    /// Sql Injection Attempt
    /// </summary>
    [EnumTextValue(@"Sql Injection Attempt")]
    SqlInjectionAttempt = 1106,
    /// <summary>
    /// Invalid Character Provided
    /// </summary>
    [EnumTextValue(@"Invalid Character")]
    InvalidCharacter = 1107,
    /// <summary>
    /// Invalid XML Provided
    /// </summary>
    [EnumTextValue(@"Invalid XML")]
    InvalidXML = 1108,
    /// <summary>
    /// XmlSchemaException
    /// </summary>
    [EnumTextValue(@"XML Schmea Validation Failed")]
    XMLSchemaValidationFailed = 1109,
    /// <summary>
    /// Invalid data type in client request
    /// </summary>
    [EnumTextValue(@"Invalid data type in client request")]
    InvalidDataType = 1110,
    /// <summary>
    /// Invalid value in SOAP message body
    /// </summary>
    [EnumTextValue(@"Invalid value in SOAP message body")]
    InvalidValue = 1111,
    /// <summary>
    /// Invalid element in SOAP message body
    /// </summary>
    [EnumTextValue(@"Invalid element in SOAP message body")]
    InvalidElement = 1112,
    /// <summary>
    /// Invaild characters in client request
    /// </summary>
    [EnumTextValue(@"Invalid characters in client request ")]
    ProhibitedCharacters = 1115,
    /// <summary>
    /// Error authenticating client request
    /// </summary>
    [EnumTextValue(@"Error authenticating client request")]
    AuthenticationFailed = 1500,
    /// <summary>
    /// Error decrypting client request
    /// </summary>
    [EnumTextValue(@"Error decrypting client request")]
    AuthenticationEncryptionInvalid = 1505,
    /// <summary>
```

```
/// Not authorized to complete client request
/// </summary>
[EnumTextValue(@"Not authorized to complete client request")]
AuthorizationDenied = 1551,
/// <summary>
/// Invalid customerName in client request
/// </summary>
[EnumTextValue(@"Invalid customerName in client request")]
InvalidCustomerName = 2000,
/// <summary>
/// Invalid customerID in client request
/// </summary>
[EnumTextValue(@"Invalid customerID in client request")]
InvalidCustomerID = 2001,
/// <summary>
/// Invalid DriverID in client request
/// </summary>
[EnumTextValue(@"Invalid DriverID in client request")]
InvalidDriverID = 2010,
/// <summary>
/// Invalid DriverName in client request
/// </summary>
[EnumTextValue(@"Invalid DriverLogon in client request")]
InvalidDriverLogon = 2011,
/// <summary>
/// Invalid DriverName in client request
/// </summary>
[EnumTextValue(@"Invalid Driver EmployeeID in client request")]
InvalidEmployeeId = 2012,
/// <summary>
/// Invalid TruckID in client request
/// </summary>
[EnumTextValue(@"Invalid TruckID in client request ")]
InvalidTruckPCID = 2020,
/// <summary>
/// Invalid Truck in client request
/// </summary>
[EnumTextValue(@"Invalid Truck in client request")]
InvalidTruckName = 2021,
/// <summary>
/// Invalid Truck in client request
/// </summary>
[EnumTextValue(@"Invalid TruckPCSerialNr in client request")]
InvalidTruckPCSerialNr = 2022,
/// <summary>
/// File data not provided in client request to send file
/// </summary>
[EnumTextValue(@"File data not provided in client request to send file")]
FileDataNotProvided = 2550,
/// <summary>
/// File data invalid in client request to send file
/// </summary>
[EnumTextValue(@"File data invalid in client request to send file")]
FileDataInvalid = 2551,
/// <summary>
/// No file name provided in client request to send file
/// </summary>
[EnumTextValue(@"No file name provided in client request to send file")]
FileNameNotProvided = 2552,
/// <summary>
/// No file mime type provided in client request to send file
/// </summary>
[EnumTextValue(@"No file mime type provided in client request to send file")]
FileMimeTypeNotProvided = 2553,
/// <summary>
/// No recipient list provided in client request
/// </summary>
[EnumTextValue(@"No recipient list provided in client request")]
RecipientsNotProvided = 3000,
/// <summary>
/// Cannot send to recipient driver in client request
/// </summary>
[EnumTextValue(@"Cannot send to recipient driver in client request")]
RecipientDriversNotAllowed = 3001,
/// <summary>
/// Cannot send to recipient truck in client request
```

```
        /// </summary>
        [EnumTextValue(@"Cannot send to recipient truck in client request")]
        RecipientTrucksNotAllowed = 3002,
        /// <summary>
        /// Invalid file id
        /// </summary>
        [EnumTextValue(@"Invalid file id ")]
        FileIDInvalid = 3500,
        /// <summary>
        /// Invalid Date range
        /// </summary>
        [EnumTextValue(@"Invalid date range ")]
        InvalidDateRange = 4000,
        /// <summary>
        /// Arguement null or missing
        /// </summary>
        [EnumTextValue(@"Argument null or missing")]
        ArgumentNullOrMissing = 1011,
        /// <summary>
        /// Invalid form id in client request
        /// </summary>
        [EnumTextValue(@"Invalid form id in client request")]
        InvalidFormId = 4500,
        /// <summary>
        /// Invalid form field specified
        /// </summary>
        [EnumTextValue(@"Invalid form field specified")]
        InvalidFormField = 4501,
        /// <summary>
        /// Invalid form definition number in client request
        /// </summary>
        [EnumTextValue(@"Invalid form definition number specified")]
        InvalidFormDefinitionNumber = 4502,
        /// <summary>
        /// Invalid form definition
        /// </summary>
        [EnumTextValue(@"Invalid form definition")]
        InvalidFormDefinition = 4503,
    }
```

Data Sourcing Strategies

To prevent disruption by both intentional and unintentional DOS type of 'attacks' your requests will be rate limited as follows:

- Inbound (from Device->DataCenter->You) data requests by Company+User+Operation to not more than 1 every 1000ms
- Outbound (from You->DataCenter->Device) is rate-limited by Company+User+Operation to not more than 1 every 300ms

Guidelines for continual polling of various data types

Type	Recommended	Maximum Frequency	Notes
DataService: InboundMessages	5 minutes	1 minutes	Drain Queue processing may be applied
DataService: InboundFiles	10 minutes	1 minutes	Drain Queue processing may be applied
DataService: TrailerEvents	varies	1 minutes	Drain Queue processing may be applied
DataService: HosTotals	1 hours	1 minutes	HosTotals can be continually calculated on the client between polling intervals Its possible to retreive only those which have been updated since the last request
DataService: HosLogs	6 hours	1 hours	Drain Queue processing may be applied
DataService: HosUnassignedDriving	30 minutes	5 minutes	
ReportsService: ALL	6 hours	1 hours	Drain Queue processing may be applied

Continual Polling pseudo code example

Drain Queue Pseudo Code Example

```
shortPollSleep = 5                                                                          //Required
sleep amount when polling again immediately
longPollSleep = Storage.GetLongPollSleep("operationName")                              //3hours - Configurable long wait
period
batchSize = Storage.GetBatchSize("operationName")                                        //1000 - Configurable batch
size per operation/request type
currentMaxIndexReceived = Storage.GetLastPollingValue("operationName")      //Storage area to keep track of last polled
value

While(true)
{
    response = X_GetByIndex(currentMaxIndexReceived, batchSize)
    currentMaxIndexReceived = MAX(response.Items.Id)                                    //Get MAX(object) for query type
(i.e. Int for ByIndex, DateTime for other types)
    Storage.SetLastPollingValue("operationName", currentMaxIndexReceived)    //Persist the value to durable storage to
allow the app to continue after a restart
    if(COUNT(response.Items) >= batchSize)                                                  //Are there more
records?
                    sleep(shortPollSleep)                                                          //Immediately
call again to get next batch to catch up as there are more 'in the queue'
    else
                    sleep(longPollSleep)
}
```

Should you have any questions regarding the services or need assistance please contact Web Service Support via Email
Feel free to contact us for assistance in getting the most out of these Services!

Documentation is provided in the following locations:

- High Level Service Operation information - On each individual service page, below each operation AND in the WSDL linked at the top of each service page via "service reference"
- Detailed Type information - Within annotations in the various XSD files linked at the top of each service page
- Other documents referenced by the service pages - White papers, etc.
- Also available is an XSD Primer for viewing the XSD files mentioned above

Tools and Utilities

Applications which can display an XSD Graphically:

- Altova XML Spy
- Liquid Studio
- Stylus Studio XSD Editor
- XSD Diagram - Free but limited (does not display enum values etc)
- Membrane SOA - WSDL and Schema diff'ing tools
- Stackoverflow List of various apps
- You may send recommendations to the development team

During development of web services it can be very valuable to issue test requests before writing any code:

A free tool called SoapUI