Part 6 defines two DataEncodings: Compact and Verbose. They have the same structure except Verbose includes fields with default values so Verbose messages are larger.

Part 14 defines a field encoding which is a special rule for both encodings that simplifies the encoding of Variants and ExtensionObjects. By default, Part 14 fields can only be decoded by inferring the schema from the JSON data or if the metadata describing the fields is available (i.e. from the DataSetMetaData messages).

There are two edge cases that create problems for the Part 14 field encoding:

1. Fields with DataType=Structure and AllowSubtypes=TRUE which implies each value could have an instance of a different subtype of a Structure.
2. Fields with DataType=BaseDataType which implies each field can have an instance with a different DataType.

In most cases, Publishers can define DataSets that avoid these edge cases. If a Publisher needs to use a DataSet with these edge cases, they can set RawData=FALSE. This means Publishers will include the UaType and UaTypeId fields in every field where the edge cases apply.

If RawData=TRUE then subscribers will not be able unambiguously decode the messages, however, they can treat the problematic fields as generic JSON data. This is not recommended. Some Publishers may prevent users from configuring DataSets that have fields with the problematic edge cases.

Note that the edge cases can appear at the top level or nested inside a Structure value. If RawData=TRUE any fields nested in Structures have the UaType and UaTypeId fields omitted.

Note that these rules do not apply to the Compact encoding which always includes the UaType and UaTypeId fields even if RawData=TRUE.

Table nnn provides examples of field encoding with the different options.

|  |  |  |
| --- | --- | --- |
| DataType | RawData=TRUE | RawData=FALSE |
| Int32 | 1234 | 1234 |
| StructureAllowSubTypes=FALSE | { "X": 1234 "Y": "Ring"} | { "X": 1234 "Y": "Ring"} |
| StructureAllowSubTypes=TRUE | { "X": 1234 "Y": "Ring"} | {"UaTypeId": "<nodeid>" "X": 1234 "Y": "Ring"} |
| BaseDataType | { "Moon"} | { "UaType": 12, "Value": "Moon"} |
| Nested StructureY has AllowSubTypes=FALSE | { "X": 1234 "Y":  { "A": "Apple" }} | { "X": 1234 "Y":  { "A": "Apple" }} |
| Nested StructureY has AllowSubTypes=TRUE | { "X": 1234 "Y":  { "A": "Apple" }} | { "X": 1234 "Y":  { "UaTypeId": "<nodeid>" "A": "Apple" }} |