

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#) X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#) X

[Sign In or Register](#) ▾

## CODEMASTERS COMMUNITY

BROWSE ACTIVITY

Forums Clubs Calendar Staff Online Users Leaderboard

Search...



[Home](#) > F1 Games > F1® 2020 Game Forum > Technical Assistance >

[All Activity](#)

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!!

### ★ F1 2020 UDP Specification

By Hoo, May 17 in Technical Assistance

Sign in to follow this

FOLLOWERS

13

1 2 3 4 5 6 [NEXT](#) ▶ Page 1 of 7 ▾

Hoo

Posted May 17



Codemasters Staff

Codemasters

163

1,199 posts

The F1 series of games support the output of certain game data across UDP connections. This data can be used to supply race information to external applications, or to drive certain hardware (e.g. motion platforms, force feedback steering wheels and LED devices).

The following information summarises these data structures so that developers of supporting hardware or software are able to configure these to work correctly with the F1 game.

If you cannot find the information that you require, or spot any issues with this specification then please let us know below.

## Packet Information

### Packet Types

Each packet can now carry different types of data rather than having one packet which contains everything. A header has been added to each packet as well so that versioning can be tracked and it will be easier for applications to check they are interpreting the incoming data in the correct way. Please note that all values are encoded using Little Endian format. All data is packed.

The following data types are used in the structures:

| Type   | Description             |
|--------|-------------------------|
| uint8  | Unsigned 8-bit integer  |
| int8   | Signed 8-bit integer    |
| uint16 | Unsigned 16-bit integer |
| int16  | Signed 16-bit integer   |
| float  | Floating point (32-bit) |
| uint64 | Unsigned 64-bit integer |

### Packet Header

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```

uint16    m_packetFormat;           // 2020
uint8     m_gameMajorVersion;      // Game major version - "X.00"
uint8     m_gameMinorVersion;      // Game minor version - "1.XX"
uint8     m_packetVersion;         // Version of this packet type, all start from 1
uint8     m_packetId;             // Identifier for the packet type, see below
uint64    m_sessionUID;            // Unique identifier for the session
float    m_sessionTime;            // Session timestamp
uint32    m_frameIdentifier;       // Identifier for the frame the data was retrieved on
uint8     m_playerCarIndex;         // Index of player's car in the array

// ADDED IN BETA 2:
uint8     m_secondaryPlayerCarIndex; // Index of secondary player's car in the array (splitscreen)
                                    // 255 if no second player
};
```

## Packet IDs

The packets IDs are as follows:

| Packet Name          | Value | Description  |
|----------------------|-------|--|
| Motion               | 0     | Contains all motion data for player's car – only sent while player is in control |
| Session              | 1     | Data about the session – track, time left  |
| Lap Data             | 2     | Data about all the lap times of cars in the session                              |
| Event                | 3     | Various notable events that happen during a session                              |
| Participants         | 4     | List of participants in the session, mostly relevant for multiplayer             |
| Car Setups           | 5     | Packet detailing car setups for cars in the race                                 |
| Car Telemetry        | 6     | Telemetry data for all cars  |
| Car Status           | 7     | Status data for all cars such as damage  |
| Final Classification | 8     | Final classification confirmation at the end of a race                           |
| Lobby Info           | 9     | Information about players in a multiplayer lobby                                 |

## Motion Packet

The motion packet gives physics data for all the cars being driven. There is additional data for the car being driven with the goal of being able to drive a motion platform setup.

*N.B. For the normalised vectors below, to convert to float values divide by 32767.0f – 16-bit signed values are used to pack the data and on the assumption that direction values are always between -1.0f and 1.0f.*

Frequency: Rate as specified in menus

Size: 1464 bytes (Packet size updated in Beta 3)

Version: 1

```

struct CarMotionData
{
    float      m_worldPositionX;        // World space X position
    float      m_worldPositionY;        // World space Y position
    float      m_worldPositionZ;        // World space Z position
    float      m_worldVelocityX;        // World space X velocity
    float      m_worldVelocityY;        // World space Y velocity
    float      m_worldVelocityZ;        // World space Z velocity
    float      m_worldAngularVelocityX; // World space X angular velocity
    float      m_worldAngularVelocityY; // World space Y angular velocity
    float      m_worldAngularVelocityZ; // World space Z angular velocity
    float      m_gyroX;                // Gyro X
    float      m_gyroY;                // Gyro Y
    float      m_gyroZ;                // Gyro Z
    float      m_accelerationX;         // Acceleration X
    float      m_accelerationY;         // Acceleration Y
    float      m_accelerationZ;         // Acceleration Z
    float      m_steer;                // Steering
    float      m_throttle;              // Throttle
    float      m_brake;                // Brake
    float      m_clutch;               // Clutch
    float      m_revolution;             // Revolution
    float      m_damages;               // Damages
    float      m_damageType;             // Damage Type
    float      m_damageLocationX;       // Damage Location X
    float      m_damageLocationY;       // Damage Location Y
    float      m_damageLocationZ;       // Damage Location Z
    float      m_damageDepth;             // Damage Depth
    float      m_damageWidth;              // Damage Width
    float      m_damageHeight;             // Damage Height
    float      m_damageArea;               // Damage Area
    float      m_damageRadius;              // Damage Radius
    float      m_damageScale;              // Damage Scale
    float      m_damageType2;              // Damage Type 2
    float      m_damageLocationX2;        // Damage Location X 2
    float      m_damageLocationY2;        // Damage Location Y 2
    float      m_damageLocationZ2;        // Damage Location Z 2
    float      m_damageDepth2;             // Damage Depth 2
    float      m_damageWidth2;              // Damage Width 2
    float      m_damageHeight2;             // Damage Height 2
    float      m_damageArea2;               // Damage Area 2
    float      m_damageRadius2;              // Damage Radius 2
    float      m_damageScale2;              // Damage Scale 2
    float      m_damageType3;              // Damage Type 3
    float      m_damageLocationX3;        // Damage Location X 3
    float      m_damageLocationY3;        // Damage Location Y 3
    float      m_damageLocationZ3;        // Damage Location Z 3
    float      m_damageDepth3;             // Damage Depth 3
    float      m_damageWidth3;              // Damage Width 3
    float      m_damageHeight3;             // Damage Height 3
    float      m_damageArea3;               // Damage Area 3
    float      m_damageRadius3;              // Damage Radius 3
    float      m_damageScale3;              // Damage Scale 3
    float      m_damageType4;              // Damage Type 4
    float      m_damageLocationX4;        // Damage Location X 4
    float      m_damageLocationY4;        // Damage Location Y 4
    float      m_damageLocationZ4;        // Damage Location Z 4
    float      m_damageDepth4;             // Damage Depth 4
    float      m_damageWidth4;              // Damage Width 4
    float      m_damageHeight4;             // Damage Height 4
    float      m_damageArea4;               // Damage Area 4
    float      m_damageRadius4;              // Damage Radius 4
    float      m_damageScale4;              // Damage Scale 4
    float      m_damageType5;              // Damage Type 5
    float      m_damageLocationX5;        // Damage Location X 5
    float      m_damageLocationY5;        // Damage Location Y 5
    float      m_damageLocationZ5;        // Damage Location Z 5
    float      m_damageDepth5;             // Damage Depth 5
    float      m_damageWidth5;              // Damage Width 5
    float      m_damageHeight5;             // Damage Height 5
    float      m_damageArea5;               // Damage Area 5
    float      m_damageRadius5;              // Damage Radius 5
    float      m_damageScale5;              // Damage Scale 5
    float      m_damageType6;              // Damage Type 6
    float      m_damageLocationX6;        // Damage Location X 6
    float      m_damageLocationY6;        // Damage Location Y 6
    float      m_damageLocationZ6;        // Damage Location Z 6
    float      m_damageDepth6;             // Damage Depth 6
    float      m_damageWidth6;              // Damage Width 6
    float      m_damageHeight6;             // Damage Height 6
    float      m_damageArea6;               // Damage Area 6
    float      m_damageRadius6;              // Damage Radius 6
    float      m_damageScale6;              // Damage Scale 6
    float      m_damageType7;              // Damage Type 7
    float      m_damageLocationX7;        // Damage Location X 7
    float      m_damageLocationY7;        // Damage Location Y 7
    float      m_damageLocationZ7;        // Damage Location Z 7
    float      m_damageDepth7;             // Damage Depth 7
    float      m_damageWidth7;              // Damage Width 7
    float      m_damageHeight7;             // Damage Height 7
    float      m_damageArea7;               // Damage Area 7
    float      m_damageRadius7;              // Damage Radius 7
    float      m_damageScale7;              // Damage Scale 7
    float      m_damageType8;              // Damage Type 8
    float      m_damageLocationX8;        // Damage Location X 8
    float      m_damageLocationY8;        // Damage Location Y 8
    float      m_damageLocationZ8;        // Damage Location Z 8
    float      m_damageDepth8;             // Damage Depth 8
    float      m_damageWidth8;              // Damage Width 8
    float      m_damageHeight8;             // Damage Height 8
    float      m_damageArea8;               // Damage Area 8
    float      m_damageRadius8;              // Damage Radius 8
    float      m_damageScale8;              // Damage Scale 8
    float      m_damageType9;              // Damage Type 9
    float      m_damageLocationX9;        // Damage Location X 9
    float      m_damageLocationY9;        // Damage Location Y 9
    float      m_damageLocationZ9;        // Damage Location Z 9
    float      m_damageDepth9;             // Damage Depth 9
    float      m_damageWidth9;              // Damage Width 9
    float      m_damageHeight9;             // Damage Height 9
    float      m_damageArea9;               // Damage Area 9
    float      m_damageRadius9;              // Damage Radius 9
    float      m_damageScale9;              // Damage Scale 9
    float      m_damageType10;             // Damage Type 10
    float      m_damageLocationX10;        // Damage Location X 10
    float      m_damageLocationY10;        // Damage Location Y 10
    float      m_damageLocationZ10;        // Damage Location Z 10
    float      m_damageDepth10;             // Damage Depth 10
    float      m_damageWidth10;             // Damage Width 10
    float      m_damageHeight10;            // Damage Height 10
    float      m_damageArea10;              // Damage Area 10
    float      m_damageRadius10;             // Damage Radius 10
    float      m_damageScale10;             // Damage Scale 10
    float      m_damageType11;             // Damage Type 11
    float      m_damageLocationX11;        // Damage Location X 11
    float      m_damageLocationY11;        // Damage Location Y 11
    float      m_damageLocationZ11;        // Damage Location Z 11
    float      m_damageDepth11;             // Damage Depth 11
    float      m_damageWidth11;             // Damage Width 11
    float      m_damageHeight11;            // Damage Height 11
    float      m_damageArea11;              // Damage Area 11
    float      m_damageRadius11;             // Damage Radius 11
    float      m_damageScale11;             // Damage Scale 11
    float      m_damageType12;             // Damage Type 12
    float      m_damageLocationX12;        // Damage Location X 12
    float      m_damageLocationY12;        // Damage Location Y 12
    float      m_damageLocationZ12;        // Damage Location Z 12
    float      m_damageDepth12;             // Damage Depth 12
    float      m_damageWidth12;             // Damage Width 12
    float      m_damageHeight12;            // Damage Height 12
    float      m_damageArea12;              // Damage Area 12
    float      m_damageRadius12;             // Damage Radius 12
    float      m_damageScale12;             // Damage Scale 12
    float      m_damageType13;             // Damage Type 13
    float      m_damageLocationX13;        // Damage Location X 13
    float      m_damageLocationY13;        // Damage Location Y 13
    float      m_damageLocationZ13;        // Damage Location Z 13
    float      m_damageDepth13;             // Damage Depth 13
    float      m_damageWidth13;             // Damage Width 13
    float      m_damageHeight13;            // Damage Height 13
    float      m_damageArea13;              // Damage Area 13
    float      m_damageRadius13;             // Damage Radius 13
    float      m_damageScale13;             // Damage Scale 13
    float      m_damageType14;             // Damage Type 14
    float      m_damageLocationX14;        // Damage Location X 14
    float      m_damageLocationY14;        // Damage Location Y 14
    float      m_damageLocationZ14;        // Damage Location Z 14
    float      m_damageDepth14;             // Damage Depth 14
    float      m_damageWidth14;             // Damage Width 14
    float      m_damageHeight14;            // Damage Height 14
    float      m_damageArea14;              // Damage Area 14
    float      m_damageRadius14;             // Damage Radius 14
    float      m_damageScale14;             // Damage Scale 14
    float      m_damageType15;             // Damage Type 15
    float      m_damageLocationX15;        // Damage Location X 15
    float      m_damageLocationY15;        // Damage Location Y 15
    float      m_damageLocationZ15;        // Damage Location Z 15
    float      m_damageDepth15;             // Damage Depth 15
    float      m_damageWidth15;             // Damage Width 15
    float      m_damageHeight15;            // Damage Height 15
    float      m_damageArea15;              // Damage Area 15
    float      m_damageRadius15;             // Damage Radius 15
    float      m_damageScale15;             // Damage Scale 15
    float      m_damageType16;             // Damage Type 16
    float      m_damageLocationX16;        // Damage Location X 16
    float      m_damageLocationY16;        // Damage Location Y 16
    float      m_damageLocationZ16;        // Damage Location Z 16
    float      m_damageDepth16;             // Damage Depth 16
    float      m_damageWidth16;             // Damage Width 16
    float      m_damageHeight16;            // Damage Height 16
    float      m_damageArea16;              // Damage Area 16
    float      m_damageRadius16;             // Damage Radius 16
    float      m_damageScale16;             // Damage Scale 16
    float      m_damageType17;             // Damage Type 17
    float      m_damageLocationX17;        // Damage Location X 17
    float      m_damageLocationY17;        // Damage Location Y 17
    float      m_damageLocationZ17;        // Damage Location Z 17
    float      m_damageDepth17;             // Damage Depth 17
    float      m_damageWidth17;             // Damage Width 17
    float      m_damageHeight17;            // Damage Height 17
    float      m_damageArea17;              // Damage Area 17
    float      m_damageRadius17;             // Damage Radius 17
    float      m_damageScale17;             // Damage Scale 17
    float      m_damageType18;             // Damage Type 18
    float      m_damageLocationX18;        // Damage Location X 18
    float      m_damageLocationY18;        // Damage Location Y 18
    float      m_damageLocationZ18;        // Damage Location Z 18
    float      m_damageDepth18;             // Damage Depth 18
    float      m_damageWidth18;             // Damage Width 18
    float      m_damageHeight18;            // Damage Height 18
    float      m_damageArea18;              // Damage Area 18
    float      m_damageRadius18;             // Damage Radius 18
    float      m_damageScale18;             // Damage Scale 18
    float      m_damageType19;             // Damage Type 19
    float      m_damageLocationX19;        // Damage Location X 19
    float      m_damageLocationY19;        // Damage Location Y 19
    float      m_damageLocationZ19;        // Damage Location Z 19
    float      m_damageDepth19;             // Damage Depth 19
    float      m_damageWidth19;             // Damage Width 19
    float      m_damageHeight19;            // Damage Height 19
    float      m_damageArea19;              // Damage Area 19
    float      m_damageRadius19;             // Damage Radius 19
    float      m_damageScale19;             // Damage Scale 19
    float      m_damageType20;             // Damage Type 20
    float      m_damageLocationX20;        // Damage Location X 20
    float      m_damageLocationY20;        // Damage Location Y 20
    float      m_damageLocationZ20;        // Damage Location Z 20
    float      m_damageDepth20;             // Damage Depth 20
    float      m_damageWidth20;             // Damage Width 20
    float      m_damageHeight20;            // Damage Height 20
    float      m_damageArea20;              // Damage Area 20
    float      m_damageRadius20;             // Damage Radius 20
    float      m_damageScale20;             // Damage Scale 20
    float      m_damageType21;             // Damage Type 21
    float      m_damageLocationX21;        // Damage Location X 21
    float      m_damageLocationY21;        // Damage Location Y 21
    float      m_damageLocationZ21;        // Damage Location Z 21
    float      m_damageDepth21;             // Damage Depth 21
    float      m_damageWidth21;             // Damage Width 21
    float      m_damageHeight21;            // Damage Height 21
    float      m_damageArea21;              // Damage Area 21
    float      m_damageRadius21;             // Damage Radius 21
    float      m_damageScale21;             // Damage Scale 21
    float      m_damageType22;             // Damage Type 22
    float      m_damageLocationX22;        // Damage Location X 22
    float      m_damageLocationY22;        // Damage Location Y 22
    float      m_damageLocationZ22;        // Damage Location Z 22
    float      m_damageDepth22;             // Damage Depth 22
    float      m_damageWidth22;             // Damage Width 22
    float      m_damageHeight22;            // Damage Height 22
    float      m_damageArea22;              // Damage Area 22
    float      m_damageRadius22;             // Damage Radius 22
    float      m_damageScale22;             // Damage Scale 22
    float      m_damageType23;             // Damage Type 23
    float      m_damageLocationX23;        // Damage Location X 23
    float      m_damageLocationY23;        // Damage Location Y 23
    float      m_damageLocationZ23;        // Damage Location Z 23
    float      m_damageDepth23;             // Damage Depth 23
    float      m_damageWidth23;             // Damage Width 23
    float      m_damageHeight23;            // Damage Height 23
    float      m_damageArea23;              // Damage Area 23
    float      m_damageRadius23;             // Damage Radius 23
    float      m_damageScale23;             // Damage Scale 23
    float      m_damageType24;             // Damage Type 24
    float      m_damageLocationX24;        // Damage Location X 24
    float      m_damageLocationY24;        // Damage Location Y 24
    float      m_damageLocationZ24;        // Damage Location Z 24
    float      m_damageDepth24;             // Damage Depth 24
    float      m_damageWidth24;             // Damage Width 24
    float      m_damageHeight24;            // Damage Height 24
    float      m_damageArea24;              // Damage Area 24
    float      m_damageRadius24;             // Damage Radius 24
    float      m_damageScale24;             // Damage Scale 24
    float      m_damageType25;             // Damage Type 25
    float      m_damageLocationX25;        // Damage Location X 25
    float      m_damageLocationY25;        // Damage Location Y 25
    float      m_damageLocationZ25;        // Damage Location Z 25
    float      m_damageDepth25;             // Damage Depth 25
    float      m_damageWidth25;             // Damage Width 25
    float      m_damageHeight25;            // Damage Height 25
    float      m_damageArea25;              // Damage Area 25
    float      m_damageRadius25;             // Damage Radius 25
    float      m_damageScale25;             // Damage Scale 25
    float      m_damageType26;             // Damage Type 26
    float      m_damageLocationX26;        // Damage Location X 26
    float      m_damageLocationY26;        // Damage Location Y 26
    float      m_damageLocationZ26;        // Damage Location Z 26
    float      m_damageDepth26;             // Damage Depth 26
    float      m_damageWidth26;             // Damage Width 26
    float      m_damageHeight26;            // Damage Height 26
    float      m_damageArea26;              // Damage Area 26
    float      m_damageRadius26;             // Damage Radius 26
    float      m_damageScale26;             // Damage Scale 26
    float      m_damageType27;             // Damage Type 27
    float      m_damageLocationX27;        // Damage Location X 27
    float      m_damageLocationY27;        // Damage Location Y 27
    float      m_damageLocationZ27;        // Damage Location Z 27
    float      m_damageDepth27;             // Damage Depth 27
    float      m_damageWidth27;             // Damage Width 27
    float      m_damageHeight27;            // Damage Height 27
    float      m_damageArea27;              // Damage Area 27
    float      m_damageRadius27;             // Damage Radius 27
    float      m_damageScale27;             // Damage Scale 27
    float      m_damageType28;             // Damage Type 28
    float      m_damageLocationX28;        // Damage Location X 28
    float      m_damageLocationY28;        // Damage Location Y 28
    float      m_damageLocationZ28;        // Damage Location Z 28
    float      m_damageDepth28;             // Damage Depth 28
    float      m_damageWidth28;             // Damage Width 28
    float      m_damageHeight28;            // Damage Height 28
    float      m_damageArea28;              // Damage Area 28
    float      m_damageRadius28;             // Damage Radius 28
    float      m_damageScale28;             // Damage Scale 28
    float      m_damageType29;             // Damage Type 29
    float      m_damageLocationX29;        // Damage Location X 29
    float      m_damageLocationY29;        // Damage Location Y 29
    float      m_damageLocationZ29;        // Damage Location Z 29
    float      m_damageDepth29;             // Damage Depth 29
    float      m_damageWidth29;             // Damage Width 29
    float      m_damageHeight29;            // Damage Height 29
    float      m_damageArea29;              // Damage Area 29
    float      m_damageRadius29;             // Damage Radius 29
    float      m_damageScale29;             // Damage Scale 29
    float      m_damageType30;             // Damage Type 30
    float      m_damageLocationX30;        // Damage Location X 30
    float      m_damageLocationY30;        // Damage Location Y 30
    float      m_damageLocationZ30;        // Damage Location Z 30
    float      m_damageDepth30;             // Damage Depth 30
    float      m_damageWidth30;             // Damage Width 30
    float      m_damageHeight30;            // Damage Height 30
    float      m_damageArea30;              // Damage Area 30
    float      m_damageRadius30;             // Damage Radius 30
    float      m_damageScale30;             // Damage Scale 30
    float      m_damageType31;             // Damage Type 31
    float      m_damageLocationX31;        // Damage Location X 31
    float      m_damageLocationY31;        // Damage Location Y 31
    float      m_damageLocationZ31;        // Damage Location Z 31
    float      m_damageDepth31;             // Damage Depth 31
    float      m_damageWidth31;             // Damage Width 31
    float      m_damageHeight31;            // Damage Height 31
    float      m_damageArea31;              // Damage Area 31
    float      m_damageRadius31;             // Damage Radius 31
    float      m_damageScale31;             // Damage Scale 31
    float      m_damageType32;             // Damage Type 32
    float      m_damageLocationX32;        // Damage Location X 32
    float      m_damageLocationY32;        // Damage Location Y 32
    float      m_damageLocationZ32;        // Damage Location Z 32
    float      m_damageDepth32;             // Damage Depth 32
    float      m_damageWidth32;             // Damage Width 32
    float      m_damageHeight32;            // Damage Height 32
    float      m_damageArea32;              // Damage Area 32
    float      m_damageRadius32;             // Damage Radius 32
    float      m_damageScale32;             // Damage Scale 32
    float      m_damageType33;             // Damage Type 33
    float      m_damageLocationX33;        // Damage Location X 33
    float      m_damageLocationY33;        // Damage Location Y 33
    float      m_damageLocationZ33;        // Damage Location Z 33
    float      m_damageDepth33;             // Damage Depth 33
    float      m_damageWidth33;             // Damage Width 33
    float      m_damageHeight33;            // Damage Height 33
    float      m_damageArea33;              // Damage Area 33
    float      m_damageRadius33;             // Damage Radius 33
    float      m_damageScale33;             // Damage Scale 33
    float      m_damageType34;             // Damage Type 34
    float      m_damageLocationX34;        // Damage Location X 34
    float      m_damageLocationY34;        // Damage Location Y 34
    float      m_damageLocationZ34;        // Damage Location Z 34
    float      m_damageDepth34;             // Damage Depth 34
    float      m_damageWidth34;             // Damage Width 34
    float      m_damageHeight34;            // Damage Height 34
    float      m_damageArea34;              // Damage Area 34
    float      m_damageRadius34;             // Damage Radius 34
    float      m_damageScale34;             // Damage Scale 34
    float      m_damageType35;             // Damage Type 35
    float      m_damageLocationX35;        // Damage Location X 35
    float      m_damageLocationY35;        // Damage Location Y 35
    float      m_damageLocationZ35;        // Damage Location Z 35
    float      m_damageDepth35;             // Damage Depth 35
    float      m_damageWidth35;             // Damage Width 35
    float      m_damageHeight35;            // Damage Height 35
    float      m_damageArea35;              // Damage Area 35
    float      m_damageRadius35;             // Damage Radius 35
    float      m_damageScale35;             // Damage Scale 35
    float      m_damageType36;             // Damage Type 36
    float      m_damageLocationX36;        // Damage Location X 36
    float      m_damageLocationY36;        // Damage Location Y 36
    float      m_damageLocationZ36;        // Damage Location Z 36
    float      m_damageDepth36;             // Damage Depth 36
    float      m_damageWidth36;             // Damage Width 36
    float      m_damageHeight36;            // Damage Height 36
    float      m_damageArea36;              // Damage Area 36
    float      m_damageRadius36;             // Damage Radius 36
    float      m_damageScale36;             // Damage Scale 36
    float      m_damageType37;             // Damage Type 37
    float      m_damageLocationX37;        // Damage Location X 37
    float      m_damageLocationY37;        // Damage Location Y 37
    float      m_damageLocationZ37;        // Damage Location Z 37
    float      m_damageDepth37;             // Damage Depth 37
    float      m_damageWidth37;             // Damage Width 37
    float      m_damageHeight37;            // Damage Height 37
    float      m_damageArea37;              // Damage Area 37
    float      m_damageRadius37;             // Damage Radius 37
    float      m_damageScale37;             // Damage Scale 37
    float      m_damageType38;             // Damage Type 38
    float      m_damageLocationX38;        // Damage Location X 38
    float      m_damageLocationY38;        // Damage Location Y 38
    float      m_damageLocationZ38;        // Damage Location Z 38
    float      m_damageDepth38;             // Damage Depth 38
    float      m_damageWidth38;             // Damage Width 38
    float      m_damageHeight38;            // Damage Height 38
    float      m_damageArea38;              // Damage Area 38
    float      m_damageRadius38;             // Damage Radius 38
    float      m_damageScale38;             // Damage Scale 38
    float      m_damageType39;             // Damage Type 39
    float      m_damageLocationX39;        // Damage Location X 39
    float      m_damageLocationY39;        // Damage Location Y 39
    float      m_damageLocationZ39;        // Damage Location Z 39
    float      m_damageDepth39;             // Damage Depth 39
    float      m_damageWidth39;             // Damage Width 39
    float      m_damageHeight39;            // Damage Height 39
    float      m_damageArea39;              // Damage Area 39
    float      m_damageRadius39;             // Damage Radius 39
    float      m_damageScale39;             // Damage Scale 39
    float      m_damageType40;             // Damage Type 40
    float      m_damageLocationX40;        // Damage Location X 40
    float      m_damageLocationY40;        // Damage Location Y 40
    float      m_damageLocationZ40;        // Damage Location Z 40
    float      m_damageDepth40;             // Damage Depth 40
    float      m_damageWidth40;             // Damage Width 40
    float      m_damageHeight40;            // Damage Height 40
    float      m_damageArea40;              // Damage Area 40
    float
```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!! [Read more...](#)

X

```

int16      m_worldForwardDirX;           // World space forward X direction (normalised)
int16      m_worldForwardDirY;           // World space forward Y direction (normalised)
int16      m_worldForwardDirZ;           // World space forward Z direction (normalised)
int16      m_worldRightDirX;            // World space right X direction (normalised)
int16      m_worldRightDirY;            // World space right Y direction (normalised)
int16      m_worldRightDirZ;            // World space right Z direction (normalised)
float       m_gForceLateral;           // Lateral G-Force component
float       m_gForceLongitudinal;        // Longitudinal G-Force component
float       m_gForceVertical;           // Vertical G-Force component
float       m_yaw;                     // Yaw angle in radians
float       m_pitch;                  // Pitch angle in radians
float       m_roll;                   // Roll angle in radians
};

struct PacketMotionData
{
    PacketHeader   m_header;             // Header

    CarMotionData  m_carMotionData[22];  // Data for all cars on track

    // Extra player car ONLY data
    float         m_suspensionPosition[4]; // Note: ALL wheel arrays have the following order:
    float         m_suspensionVelocity[4]; // RL, RR, FL, FR
    float         m_suspensionAcceleration[4]; // RL, RR, FL, FR
    float         m_wheelSpeed[4];          // Speed of each wheel
    float         m_wheelSlip[4];           // Slip ratio for each wheel
    float         m_localVelocityX;        // Velocity in Local space
    float         m_localVelocityY;        // Velocity in Local space
    float         m_localVelocityZ;        // Velocity in Local space
    float         m_angularVelocityX;       // Angular velocity x-component
    float         m_angularVelocityY;       // Angular velocity y-component
    float         m_angularVelocityZ;       // Angular velocity z-component
    float         m_angularAccelerationX; // Angular velocity x-component
    float         m_angularAccelerationY; // Angular velocity y-component
    float         m_angularAccelerationZ; // Angular velocity z-component
    float         m_frontWheelsAngle;      // Current front wheels angle in radians
};

```

## Session Packet

The session packet includes details about the current session in progress.

Frequency: 2 per second

Size: 251 bytes (Packet size updated in Beta 3)

Version: 1

```

struct MarshalZone
{
    float  m_zoneStart; // Fraction (0..1) of way through the lap the marshal zone starts
    int8   m_zoneFlag;  // -1 = invalid/unknown, 0 = none, 1 = green, 2 = blue, 3 = yellow, 4 = red
};

struct WeatherForecastSample
{
    uint8   m_sessionType;           // 0 = unknown, 1 = P1, 2 = P2, 3 = P3, 4 = Short P, 5
                                      // 6 = Q2, 7 = Q3, 8 = Short Q, 9 = OSQ, 10 = R, 11 = R
                                      // 12 = Time Trial
    uint8   m_timeOffset;           // Time in minutes the forecast is for
    uint8   m_weather;              // Weather - 0 = clear, 1 = light cloud, 2 = overcast
                                      // 3 = light rain, 4 = heavy rain, 5 = storm
    int8    m_trackTemperature;     // Track temp. in degrees celsius
    int8    m_airTemperature;       // Air temp. in degrees celsius
};

```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```

uint8          m_weather;                      // weather - 0 = clear, 1 = light cloud, 2 = overcast
// 3 = light rain, 4 = heavy rain, 5 = storm
int8           m_trackTemperature;            // Track temp. in degrees celsius
int8           m_airTemperature;              // Air temp. in degrees celsius
uint8          m_totalLaps;                  // Total number of laps in this race
uint16         m_trackLength;                // Track length in metres
uint8           m_sessionType;                // 0 = unknown, 1 = P1, 2 = P2, 3 = P3, 4 = Short P
// 5 = Q1, 6 = Q2, 7 = Q3, 8 = Short Q, 9 = OSQ
// 10 = R, 11 = R2, 12 = Time Trial
int8           m_trackId;                   // -1 for unknown, 0-21 for tracks, see appendix
uint8          m_formula;                   // Formula, 0 = F1 Modern, 1 = F1 Classic, 2 = F2,
// 3 = F1 Generic
uint16         m_sessionTimeLeft;             // Time left in session in seconds
uint16         m_sessionDuration;            // Session duration in seconds
uint8          m_pitSpeedLimit;              // Pit speed limit in kilometres per hour
uint8          m_gamePaused;                 // Whether the game is paused
uint8          m_isSpectating;               // Whether the player is spectating
uint8          m_spectatorCarIndex;           // Index of the car being spectated
uint8          m_sliProNativeSupport;        // SLI Pro support, 0 = inactive, 1 = active
uint8          m_numMarshalZones;            // Number of marshal zones to follow
MarshalZone   m_marshallZones[21];          // List of marshal zones - max 21
uint8          m_safetyCarStatus;             // 0 = no safety car, 1 = full safety car
// 2 = virtual safety car
uint8          m_networkGame;                // 0 = offline, 1 = online
uint8          m_numWeatherForecastSamples; // Number of weather samples to follow
WeatherForecastSample m_weatherForecastSamples[20]; // Array of weather forecast samples
};


```

## Lap Data Packet

The lap data packet gives details of all the cars in the session.

Frequency: Rate as specified in menus

Size: 1190 bytes (Struct updated in Beta 3)

Version: 1

```

struct LapData
{
    float      m_lastLapTime;                // Last lap time in seconds
    float      m_currentLapTime;              // Current time around the lap in seconds

    //UPDATED in Beta 3:
    uint16    m_sector1TimeInMS;             // Sector 1 time in milliseconds
    uint16    m_sector2TimeInMS;             // Sector 2 time in milliseconds
    float     m_bestLapTime;                 // Best lap time of the session in seconds
    uint8     m_bestLapNum;                  // Lap number best time achieved on
    uint16    m_bestLapSector1TimeInMS;       // Sector 1 time of best lap in the session in milliseconds
    uint16    m_bestLapSector2TimeInMS;       // Sector 2 time of best lap in the session in milliseconds
    uint16    m_bestLapSector3TimeInMS;       // Sector 3 time of best lap in the session in milliseconds
    uint16    m_bestOverallSector1TimeInMS; // Best overall sector 1 time of the session in milliseconds
    uint8     m_bestOverallSector1LapNum;     // Lap number best overall sector 1 time achieved on
    uint16    m_bestOverallSector2TimeInMS; // Best overall sector 2 time of the session in milliseconds
    uint8     m_bestOverallSector2LapNum;     // Lap number best overall sector 2 time achieved on
    uint16    m_bestOverallSector3TimeInMS; // Best overall sector 3 time of the session in milliseconds
    uint8     m_bestOverallSector3LapNum;     // Lap number best overall sector 3 time achieved on

    float      m_lapDistance;                // Distance vehicle is around current lap in metres - could
// be negative if line hasn't been crossed yet
    float      m_totalDistance;              // Total distance travelled in session in metres - could
// be negative if line hasn't been crossed yet
    float      m_safetyCarDelta;              // Delta in seconds for safety car
    uint8     m_carPosition;                // Car race position
};


```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```

uint8 m_penalties;           // ACCUMULATED TIME PENALTIES IN SECONAS TO BE ADDED
uint8 m_gridPosition;        // GRID POSITION THE VEHICLE STARTED THE RACE IN
uint8 m_driverStatus;        // STATUS OF DRIVER - 0 = IN GARAGE, 1 = FLYING LAP
                             // 2 = IN LAP, 3 = OUT LAP, 4 = ON TRACK
uint8 m_resultStatus;        // RESULT STATUS - 0 = INVALID, 1 = INACTIVE, 2 = ACTIVE
                             // 3 = FINISHED, 4 = DISQUALIFIED, 5 = NOT CLASSIFIED
                             // 6 = RETIRED
};

struct PacketLapData
{
    PacketHeader m_header;      // HEADER
    LapData m_lapData[22];       // LAP DATA FOR ALL CARS ON TRACK
};

```

## Event Packet

This packet gives details of events that happen during the course of a session.

Frequency: When the event occurs

Size: 35 bytes (Packet size updated in Beta 3)

Version: 1

```

// The event details packet is different for each type of event.
// Make sure only the correct type is interpreted.
union EventDataDetails
{
    struct
    {
        uint8 vehicleIdx; // Vehicle index of car achieving fastest lap
        float lapTime;    // Lap time is in seconds
    } FastestLap;

    struct
    {
        uint8 vehicleIdx; // Vehicle index of car retiring
    } Retirement;

    struct
    {
        uint8 vehicleIdx; // Vehicle index of team mate
    } TeamMateInPits;

    struct
    {
        uint8 vehicleIdx; // Vehicle index of the race winner
    } RaceWinner;

    struct
    {
        uint8 penaltyType;          // Penalty type - see Appendices
        uint8 infringementType;     // Infringement type - see Appendices
        uint8 vehicleIdx;           // Vehicle index of the car the penalty is applied to
        uint8 otherVehicleIdx;      // Vehicle index of the other car involved
        uint8 time;                // Time gained, or time spent doing action in seconds
        uint8 lapNum;               // Lap the penalty occurred on
        uint8 placesGained;         // Number of places gained by this
    } Penalty;
};

struct
{

```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```
struct PacketEventData
{
    PacketHeader m_header;           // Header

    uint8 m_eventStringCode[4]; // Event string code, see below
    EventDataDetails m_eventDetails; // Event details - should be interpreted differently
                                    // for each type
};
```

## Event String Codes

| Event                | Code   | Description                                    |
|----------------------|--------|--|
| Session Started      | “SSTA” | Sent when the session starts                   |
| Session Ended        | “SEND” | Sent when the session ends                     |
| Fastest Lap          | “FTLP” | When a driver achieves the fastest lap         |
| Retirement           | “RTMT” | When a driver retires                          |
| DRS enabled          | “DRSE” | Race control have enabled DRS                  |
| DRS disabled         | “DRSD” | Race control have disabled DRS                 |
| Team mate in pits    | “TMPT” | Your team mate has entered the pits            |
| Chequered flag       | “CHQF” | The chequered flag has been waved              |
| Race Winner          | “RCWN” | The race winner is announced                   |
| Penalty Issued       | “PENA” | A penalty has been issued – details in event   |
| Speed Trap Triggered | “SPTP” | Speed trap has been triggered by fastest speed |

## Participants Packet

This is a list of participants in the race. If the vehicle is controlled by AI, then the name will be the driver name. If this is a multiplayer game, the names will be the Steam Id on PC, or the LAN name if appropriate.

N.B. on Xbox One, the names will always be the driver name, on PS4 the name will be the LAN name if playing a LAN game, otherwise it will be the driver name.

The array should be indexed by vehicle index.

Frequency: Every 5 seconds

Size: 1213 bytes (Packet size updated in Beta 3)

Version: 1

```
struct ParticipantData
{
    uint8 m_aiControlled;           // Whether the vehicle is AI (1) or Human (0) controlled
    uint8 m_driverId;              // Driver id - see appendix
    uint8 m_teamId;                // Team id - see appendix
    uint8 m_raceNumber;             // Race number of the car
```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```
};

struct PacketParticipantsData
{
    PacketHeader m_header; // Header

    uint8 m_numActiveCars; // Number of active cars in the data - should match number of
                           // cars on HUD
    ParticipantData m_participants[22];
};
```

## Car Setups Packet

This packet details the car setups for each vehicle in the session. Note that in multiplayer games, other player cars will appear as blank, you will only be able to see your car setup and AI cars.

Frequency: 2 per second

Size: 1102 bytes (Packet size updated in Beta 3)

Version: 1

```
struct CarSetupData
{
    uint8 m_frontWing; // Front wing aero
    uint8 m_rearWing; // Rear wing aero
    uint8 m_onThrottle; // Differential adjustment on throttle (percentage)
    uint8 m_offThrottle; // Differential adjustment off throttle (percentage)
    float m_frontCamber; // Front camber angle (suspension geometry)
    float m_rearCamber; // Rear camber angle (suspension geometry)
    float m_frontToe; // Front toe angle (suspension geometry)
    float m_rearToe; // Rear toe angle (suspension geometry)
    uint8 m_frontSuspension; // Front suspension
    uint8 m_rearSuspension; // Rear suspension
    uint8 m_frontAntiRollBar; // Front anti-roll bar
    uint8 m_rearAntiRollBar; // Front anti-roll bar
    uint8 m_frontSuspensionHeight; // Front ride height
    uint8 m_rearSuspensionHeight; // Rear ride height
    uint8 m_brakePressure; // Brake pressure (percentage)
    uint8 m_brakeBias; // Brake bias (percentage)
    float m_rearLeftTyrePressure; // Rear left tyre pressure (PSI)
    float m_rearRightTyrePressure; // Rear right tyre pressure (PSI)
    float m_frontLeftTyrePressure; // Front Left tyre pressure (PSI)
    float m_frontRightTyrePressure; // Front right tyre pressure (PSI)
    uint8 m_ballast; // Ballast
    float m_fuelLoad; // Fuel Load
};

struct PacketCarSetupData
{
    PacketHeader m_header; // Header

    CarSetupData m_carSetups[22];
};
```

## Car Telemetry Packet

This packet details telemetry for all the cars in the race. It details various values that would be recorded on the car such as speed, throttle application, DRS etc.

Frequency: Rate as specified in menus

Size: 1307 bytes (Packet size updated in Beta 3)

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```

uint16 m_speed;                                // Speed of car in kilometres per hour
float m_throttle;                             // Amount of throttle applied (0.0 to 1.0)
float m_steer;                               // Steering (-1.0 (full lock left) to 1.0 (full lock right))
float m_brake;                               // Amount of brake applied (0.0 to 1.0)
uint8 m_clutch;                             // Amount of clutch applied (0 to 100)
int8 m_gear;                                // Gear selected (1-8, N=0, R=-1)
uint16 m_engineRPM;                          // Engine RPM
uint8 m_drs;                                 // 0 = off, 1 = on
uint8 m_revLightsPercent;                    // Rev Lights indicator (percentage)
uint16 m_brakesTemperature[4];                // Brakes temperature (celsius)
uint8 m_tyresSurfaceTemperature[4];           // Tyres surface temperature (celsius)
uint8 m_tyresInnerTemperature[4];              // Tyres inner temperature (celsius)
uint16 m_engineTemperature;                  // Engine temperature (celsius)
float m_tyresPressure[4];                     // Tyres pressure (PSI)
uint8 m_surfaceType[4];                      // Driving surface, see appendices
};

struct PacketCarTelemetryData
{
    PacketHeader m_header;                      // Header

    CarTelemetryData m_carTelemetryData[22];

    uint32 m_buttonStatus;                     // Bit flags specifying which buttons are being pressed
                                                // currently - see appendices

    // Added in Beta 3:
    uint8 m_mfdPanelIndex;                   // Index of MFD panel open - 255 = MFD closed
                                                // Single player, race - 0 = Car setup, 1 = Pits
                                                // 2 = Damage, 3 = Engine, 4 = Temperatures
                                                // May vary depending on game mode
    uint8 m_mfdPanelIndexSecondaryPlayer;    // See above
    int8 m_suggestedGear;                   // Suggested gear for the pLayer (1-8)
                                                // 0 if no gear suggested
};

```

## Car Status Packet

This packet details car statuses for all the cars in the race. It includes values such as the damage readings on the car.

Frequency: Rate as specified in menus

Size: 1344 bytes (Packet updated in Beta 3)

Version: 1

```

struct CarStatusData
{
    uint8 m_tractionControl;                 // 0 (off) - 2 (high)
    uint8 m_antiLockBrakes;                  // 0 (off) - 1 (on)
    uint8 m_fuelMix;                        // Fuel mix - 0 = Lean, 1 = standard, 2 = rich, 3 = max
    uint8 m_frontBrakeBias;                 // Front brake bias (percentage)
    uint8 m_pitLimiterStatus;                // Pit Limiter status - 0 = off, 1 = on
    float m_fuelInTank;                     // Current fuel mass
    float m_fuelCapacity;                   // Fuel capacity
    float m_fuelRemainingLaps;               // Fuel remaining in terms of Laps (value on MFD)
    uint16 m_maxRPM;                        // Cars max RPM, point of rev limiter
    uint16 m_idleRPM;                        // Cars idle RPM
    uint8 m_maxGears;                       // Maximum number of gears
    uint8 m_drsAllowed;                     // 0 = not allowed, 1 = allowed, -1 = unknown

    // Added in Beta3:
    uint16 m_drsActivationDistance;          // 0 = DRS not available, non-zero - DRS will be available
                                                // in [X] metres
};

```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```

// F1 Classic - 9 = dry, 10 = wet
// F2 - 11 = super soft, 12 = soft, 13 = medium, 14 = hard
// 15 = wet
uint8 m_visualTyreCompound; // F1 visual (can be different from actual compound)
                           // 16 = soft, 17 = medium, 18 = hard, 7 = inter, 8 = wet
                           // F1 Classic - same as above
                           // F2 - same as above
uint8 m_tyresAgeLaps; // Age in Laps of the current set of tyres
uint8 m_tyresDamage[4]; // Tyre damage (percentage)
uint8 m_frontLeftWingDamage; // Front Left wing damage (percentage)
uint8 m_frontRightWingDamage; // Front right wing damage (percentage)
uint8 m_rearWingDamage; // Rear wing damage (percentage)

// Added Beta 3:
uint8 m_drsFault; // Indicator for DRS fault, 0 = OK, 1 = fault

uint8 m_engineDamage; // Engine damage (percentage)
uint8 m_gearBoxDamage; // Gear box damage (percentage)
int8 m_vehicleFiaFlags; // -1 = invalid/unknown, 0 = none, 1 = green
                        // 2 = blue, 3 = yellow, 4 = red
float m_ersStoreEnergy; // ERS energy store in Joules
uint8 m_ersDeployMode; // ERS deployment mode, 0 = none, 1 = medium
                      // 2 = overtake, 3 = hotlap
float m_ersHarvestedThisLapMGUK; // ERS energy harvested this Lap by MGU-K
float m_ersHarvestedThisLapMGUH; // ERS energy harvested this Lap by MGU-H
float m_ersDeployedThisLap; // ERS energy deployed this Lap
};

struct PacketCarStatusData
{
    PacketHeader m_header; // Header
    CarStatusData m_carStatusData[22];
};

```

## Final Classification Packet

This packet details the final classification at the end of the race, and the data will match with the post race results screen. This is especially useful for multiplayer games where it is not always possible to send lap times on the final frame because of network delay.

Frequency: Once at the end of a race

Size: 839 bytes (Packet size updated in Beta 3)

Version: 1

```

struct FinalClassificationData
{
    uint8 m_position; // Finishing position
    uint8 m_numLaps; // Number of laps completed
    uint8 m_gridPosition; // Grid position of the car
    uint8 m_points; // Number of points scored
    uint8 m_numPitStops; // Number of pit stops made
    uint8 m_resultStatus; // Result status - 0 = invalid, 1 = inactive, 2 = active
                          // 3 = finished, 4 = disqualified, 5 = not classified
                          // 6 = retired
    float m_bestLapTime; // Best lap time of the session in seconds
    double m_totalRaceTime; // Total race time in seconds without penalties
    uint8 m_penaltiesTime; // Total penalties accumulated in seconds
    uint8 m_numPenalties; // Number of penalties applied to this driver
    uint8 m_numTyreStints; // Number of tyres stints up to maximum
    uint8 m_tyreStintsActual[8]; // Actual tyres used by this driver
    uint8 m_tyreStintsVisual[8]; // Visual tyres used by this driver
};

```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```
    uint8          m_numCars;           // Number of cars in the final classification
    FinalClassificationData  m_classificationData[22];
};
```

## Lobby Info Packet

This packet details the players currently in a multiplayer lobby. It details each player's selected car, any AI involved in the game and also the ready status of each of the participants.

Frequency: Two every second when in the lobby

Size: 1169 bytes (Packet size updated in Beta 3)

Version: 1

```
struct LobbyInfoData
{
    uint8    m_aiControlled;           // Whether the vehicle is AI (1) or Human (0) controlled
    uint8    m_teamId;                // Team id - see appendix (255 if no team currently selected)
    uint8    m_nationality;           // Nationality of the driver
    char     m_name[48];              // Name of participant in UTF-8 format - null terminated
                                         // Will be truncated with ... (U+2026) if too long
    uint8    m_readyStatus;           // 0 = not ready, 1 = ready, 2 = spectating
};

struct PacketLobbyInfoData
{
    PacketHeader   m_header;          // Header

    // Packet specific data
    uint8          m_numPlayers;      // Number of players in the Lobby data
    LobbyInfoData  m_lobbyPlayers[22];
};
```

## Restricted data (Your Telemetry setting)

There is some data in the UDP that you may not want other players seeing if you are in a multiplayer game. This is controlled by the "Your Telemetry" setting in the Telemetry options. The options are:

- Restricted (Default) – other players viewing the UDP data will not see values for your car
- Public – all other players can see all the data for your car

Note: You can always see the data for the car you are driving regardless of the setting.

The following data items are set to zero if the player driving the car in question has their "Your Telemetry" set to "Restricted":

### Car status packet

- m\_fuelInTank
- m\_fuelCapacity
- m\_fuelMix
- m\_fuelRemainingLaps
- m\_frontBrakeBias
- m\_frontLeftWingDamage
- m\_frontRightWingDamage
- m\_rearWingDamage
- m\_engineDamage
- m\_gearBoxDamage
- m\_tyresWear (All four wheels)
- m\_tyresDamage (All four wheels)
- m\_ersDeployMode
- m\_ersStoreEnergy
- m\_ersDeployedThisLap
- m\_ersHarvestedThisLapMGUK
- m\_ersHarvestedThisLapMGUH
- m\_tyresAgeLaps

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

x

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

x

Hoo

Posted May 17 (edited)

x



Codemasters Staff



+ 163

1,199 posts

## FAQS

### How do I enable the UDP Telemetry Output?

In F1 2020, UDP telemetry output is controlled via the in-game menus. To enable this, enter the options menu from the main menu (triangle / Y), then enter the settings menu - the UDP option will be at the bottom of the list. From there you will be able to enable / disable the UDP output, configure the IP address and port for the receiving application, toggle broadcast mode and set the send rate. Broadcast mode transmits the data across the network subnet to allow multiple devices on the same subnet to be able to receive this information. When using broadcast mode it is not necessary to set a target IP address, just a target port for applications to listen on.

*Advanced PC Users:* You can additionally edit the game's configuration XML file to configure UDP output. The file is located here (after an initial boot of the game):

```
... \Documents\My Games\<game_folder>\hardwaresettings\hardware_settings_config.xml
```

You should see the tag:

```
<motion> ...
  <udp enabled="false" broadcast="false" ip="127.0.0.1" port="20777" sendRate="20" format="2020" yourTel ...
  ...
</motion>
```

Here you can set the values manually. Note that any changes made within the game when it is running will overwrite any changes made manually. Note the enabled flag is now a state.

### What has changed since last year?

- F1 2020 sees the following changes to the UDP specification:
- Penalties have been added as a new Event Type
- Weather forecast data is now available in session packets for upcoming sessions
- Reduced the size of the surface and inner tyre temperature fields in the Car Telemetry to reduce overall packet size as more vehicles need to be added
- My Team allows an extra team to race – this means that all the places in the packets where 20 cars were used, 22 are now needed. N.B. this will not be fixed in old formats (2019, 2018, legacy) – if you are in the “My Team” career mode with any format other than 2020 specified, no data will be output. All other game modes will function as before
- Added Vietnamese and Barbadian nationalities
- Added Final Classification packet for end of race results
- Made m\_gridPosition the actual numerical position, not 0-based (only in 2020 data)
- Added Lobby Info packet to send data to UDP when in a multiplayer lobby
- Added number of laps the current set of tyres have been used for in status packet
- Split the tyre pressures in Car Setups packet into RL, RR, FL, FR to reflect the game changes
- Added secondary player car index to packet headers for splitscreen
- Added MFD Panel Index to the car telemetry packet for both players
- Added best sector times and lap numbers to the laps packet as they cannot always be recorded correctly (e.g. fast-forwarding time) – N.B. the order has been slightly rearranged to tidy up
- Changed all sector times in lap data packet to be in milliseconds (uint16 instead of float) to reduce packet size so other things can be added
- Added indicator for DRS faults to the car status packet
- Updated ERS mode values
- Speed trap triggered event added
- Added DRS activation distance to indicate whether the car has passed detection and DRS will be available in the subsequent activation zone
- Suggested gear added to the car telemetry packet

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#) X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#) X

- 0 – Rear Left (RL)
- 1 – Rear Right (RR)
- 2 – Front Left (FL)
- 3 – Front Right (FR)

## Do the vehicle indices change?

During a session, each car is assigned a vehicle index. This will not change throughout the session and all the arrays that are sent use this vehicle index to dereference the correct piece of data.

## What encoding format is used?

All values are encoded using Little Endian format.

## Are the data structures packed?

Yes, all data is packed, there is no padding used.

## Will there always be 20 cars in the data structures?

No, for F1 2020, there is a new feature called “My Team” which allows an extra team to be present on the grid. This means that all previous places where 20 cars were used, 22 is now the maximum. If “My Team” is not active however, most games modes will act as before and have a maximum of 20. Note that if your UDP format is 2019, 2018 or legacy and you are in “My Team” career mode, no UDP output will be produced because of this limitation.

There is still the data item called `m_numActiveCars` in the participants packet which tells you how many cars are active in the race. However, you should check the individual result status of each car in the lap data to see if that car is actively providing data. If it is not “Invalid” or “Inactive” then the corresponding vehicle index has valid data.

## How often are updated packets sent?

For the packets which get updated at “Rate as specified in the menus” you can be guaranteed that on the frame that these get sent they will all get sent together and will never be separated across frames. This of course relies on the reliability of your network as to whether they are received correctly as everything is sent via UDP. Other packets that get sent at specific rates can arrive on any frame.

If you are connected to the game when it starts transmitting the first frame will contain the following information to help initialise data structures on the receiving application:

Packets sent on Frame 1: (All packets sent on this frame have “Session timestamp” 0.000)

- Session
- Participants
- Car Setups
- Lap Data
- Motion Data
- Car Telemetry
- Car Status

As an example, assuming that you are running at 60Hz with 60Hz update rate selected in the menus then you would expect to see the following packets and timestamps:

Packets sent on Frame 2: (All packets sent on this frame have “Session timestamp” 0.016)

- Lap Data
- Motion Data
- Car Telemetry
- Car Status

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!! [Read more...](#)

X

- Car Setups (since 2 updates per second)
- Lap Data
- Motion Data
- Car Telemetry
- Car Status

## Will my old app still work with F1 2020?

F1 2020 uses a new format for the UDP data. However, earlier formats of the data are still supported so that most older apps implemented using the previous data formats should work with little or no change from the developer. To use the old formats, please enter the UDP options menu and set “UDP Format” to either “F1 2019”, “F1 2018” or “Legacy” (for F1 2017 and earlier).

Specifications for the legacy format can be seen here: <http://forums.codemasters.com/discussion/53139/f1-2017-d-box-and-udp-output-specification/p1>.

Specifications for the F1 2018 format can be seen here: <https://forums.codemasters.com/topic/30601-f1-2018-udp-specification/>.

## How do I enable D-BOX output?

D-BOX output is currently supported on the PC platform. In F1 2020, the D-BOX activation can be controlled via the menus. Navigate to Game Options->Settings->UDP Telemetry Settings->D-BOX to activate this on your system.

*Advanced PC Users:* It is possible to control D-BOX by editing the games’ configuration XML file. The file is located here (after an initial boot of the game):

```
...\\Documents\\My Games\\<game_folder>\\hardware\\hardware_settings_config.xml
```

You should see the tag:

```
<motion>
  <dbox enabled="false" />
  ...
</motion>
```

Set the “enabled” value to “true” to allow the game to output to your D-BOX motion platform. Note that any changes made within the game when it is running will overwrite any changes made manually.

## How can I disable in-game support for LED device?

The F1 game has native support for some of the basic features supported by some external LED devices, such as the *Leo Bodnar SLI Pro* and the *Fanatec* steering wheels. To avoid conflicts between Codemasters’ implementation and any third-party device managers on the PC platform it may be necessary to disable the native support. This is done using the following `led_display` flags in the `hardware_settings_config.xml`. The file is located here (after an initial boot of the game):

```
...\\Documents\\My Games\\<game_folder>\\hardware\\hardware_settings_config.xml
```

The flags to enable/disable LED output are:

```
<led_display fanatecNativeSupport="true" sliProNativeSupport="true" />
```

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

Please note there is an additional flag to manually control the LED brightness on the SLI Pro:

```
<led_display sliProForceBrightness="127" />
```

This option (using value in the range 0-255) will be ignored when setting the `sliProNativeSupport` flag to "false".

Also note it is now possible to edit these values on the fly via the Game Options->Settings->UDP Telemetry Settings menu.

## Can I configure the UDP output using an XML File?

PC users can edit the game's configuration XML file to configure UDP output. The file is located here (after an initial boot of the game):

```
...\\Documents\\My Games\\<game_folder>\\hardwaresettings\\hardware_settings_config.xml
```

You should see the tag:

```
<motion>
  ...
<udp enabled="false" broadcast="false" ip="127.0.0.1" port="20777" sendRate="20" format="2020" yourTel
  ...
</motion>
```

Here you can set the values manually. Note that any changes made within the game when it is running will overwrite any changes made manually.

Edited June 24 by Hoo

Updated list of changes for this year

Hoo

Posted May 17



Codemasters Staff



+ 163

1,199 posts

## Appendices

Here are the values used for the team ID, driver ID and track ID parameters.

N.B. Driver IDs in network games differ from the actual driver IDs. All the IDs of human players start at 100 and are unique within the game session, but don't directly correlate to the player.

### Team IDs

| ID | Team            | ID | Team                  | ID | Team                      |
|----|-----------------|----|-----------------------|----|---------------------------|
| 0  | Mercedes        | 21 | Red Bull 2010         | 42 | Art GP '19                |
| 1  | Ferrari         | 22 | Ferrari 1976          | 43 | Campos '19                |
| 2  | Red Bull Racing | 23 | ART Grand Prix        | 44 | Carlin '19                |
| 3  | Williams        | 24 | Campos Vexatec Racing | 45 | Sauber Junior Charouz '19 |
| 4  | Racing Point    | 25 | Carlin                | 46 | Dams '19                  |
|    |                 |    |                       |    |                           |

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

|    |               |    |                |     |               |
|----|---------------|----|----------------|-----|---------------|
| 7  | Haas          | 28 | Russian Time   | 49  | Prema '19     |
| 8  | McLaren       | 29 | MP Motorsport  | 50  | Trident '19   |
| 9  | Alfa Romeo    | 30 | Pertamina      | 51  | Arden '19     |
| 10 | McLaren 1988  | 31 | McLaren 1990   | 53  | Benetton 1994 |
| 11 | McLaren 1991  | 32 | Trident        | 54  | Benetton 1995 |
| 12 | Williams 1992 | 33 | BWT Arden      | 55  | Ferrari 2000  |
| 13 | Ferrari 1995  | 34 | McLaren 1976   | 56  | Jordan 1991   |
| 14 | Williams 1996 | 35 | Lotus 1972     |     |               |
| 15 | McLaren 1998  | 36 | Ferrari 1979   |     |               |
| 16 | Ferrari 2002  | 37 | McLaren 1982   |     |               |
| 17 | Ferrari 2004  | 38 | Williams 2003  |     |               |
| 18 | Renault 2006  | 39 | Brawn 2009     |     |               |
| 19 | Ferrari 2007  | 40 | Lotus 1978     |     |               |
| 20 | McLaren 2008  | 41 | F1 Generic car | 255 | My Team       |

## Driver IDs

| ID | Driver           | ID | Driver           | ID | Driver              |
|----|------------------|----|------------------|----|---------------------|
| 0  | Carlos Sainz     | 37 | Peter Belousov   | 70 | Rashid Nair         |
| 1  | Daniil Kvyat     | 38 | Klimek Michalski | 71 | Jack Tremblay       |
| 2  | Daniel Ricciardo | 39 | Santiago Moreno  | 74 | Antonio Giovinazzi  |
| 6  | Kimi Räikkönen   | 40 | Benjamin Coppens | 75 | Robert Kubica       |
| 7  | Lewis Hamilton   | 41 | Noah Visser      | 78 | Nobuharu Matsushita |
| 9  | Max Verstappen   | 42 | Gert Waldmuller  | 79 | Nikita Mazepin      |
| 10 | Nico Hulkenburg  | 43 | Julian Quesada   | 80 | Guanya Zhou         |
| 11 | Kevin Magnussen  | 44 | Daniel Jones     | 81 | Mick Schumacher     |
| 12 | Romain Grosjean  | 45 | Artem Markelov   | 82 | Callum Ilott        |
| 13 | Sebastian Vettel | 46 | Tadasuke Makino  | 83 | Juan Manuel Correa  |
| 14 | Sergio Perez     | 47 | Sean Gelael      | 84 | Jordan King         |

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

|    |                    |    |                     |    |                 |
|----|--------------------|----|---------------------|----|-----------------|
| 19 | Lance Stroll       | 50 | George Russell      | 87 | Anthoine Hubert |
| 20 | Arron Barnes       | 51 | Maximilian Günther  | 88 | Guiliano Alesi  |
| 21 | Martin Giles       | 52 | Nirei Fukuzumi      | 89 | Ralph Boschung  |
| 22 | Alex Murray        | 53 | Luca Ghiotto        |    |                 |
| 23 | Lucas Roth         | 54 | Lando Norris        |    |                 |
| 24 | Igor Correia       | 55 | Sérgio Sette Câmara |    |                 |
| 25 | Sophie Levasseur   | 56 | Louis Delétraz      |    |                 |
| 26 | Jonas Schiffer     | 57 | Antonio Fuoco       |    |                 |
| 27 | Alain Forest       | 58 | Charles Leclerc     |    |                 |
| 28 | Jay Letourneau     | 59 | Pierre Gasly        |    |                 |
| 29 | Esto Saari         | 62 | Alexander Albon     |    |                 |
| 30 | Yasar Atiyeh       | 63 | Nicholas Latifi     |    |                 |
| 31 | Callisto Calabresi | 64 | Dorian Boccolacci   |    |                 |
| 32 | Naota Izum         | 65 | Niko Kari           |    |                 |
| 33 | Howard Clarke      | 66 | Roberto Merhi       |    |                 |
| 34 | Wilheim Kaufmann   | 67 | Arjun Maini         |    |                 |
| 35 | Marie Laursen      | 68 | Alessio Lorandi     |    |                 |
| 36 | Flavio Nieves      | 69 | Ruben Meijer        |    |                 |

## Track IDs

| ID | Track            |
|----|------------------|
| 0  | Melbourne        |
| 1  | Paul Ricard      |
| 2  | Shanghai         |
| 3  | Sakhir (Bahrain) |
| 4  | Catalunya        |
| 5  | Monaco           |
| 6  | Montreal         |

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

|    |                   |
|----|-------------------|
| 9  | Hungaroring       |
| 10 | Spa               |
| 11 | Monza             |
| 12 | Singapore         |
| 13 | Suzuka            |
| 14 | Abu Dhabi         |
| 15 | Texas             |
| 16 | Brazil            |
| 17 | Austria           |
| 18 | Sochi             |
| 19 | Mexico            |
| 20 | Baku (Azerbaijan) |
| 21 | Sakhir Short      |
| 22 | Silverstone Short |
| 23 | Texas Short       |
| 24 | Suzuka Short      |
| 25 | Hanoi             |
| 26 | Zandvoort         |

## Nationality IDs

| ID | Nationality | ID | Nationality | ID | Nationality |
|----|-------------|----|-------------|----|-------------|
| 1  | American    | 31 | Greek       | 61 | Panamanian  |
| 2  | Argentinean | 32 | Guatemalan  | 62 | Paraguayan  |
| 3  | Australian  | 33 | Honduran    | 63 | Peruvian    |
| 4  | Austrian    | 34 | Hong Konger | 64 | Polish      |
| 5  | Azerbaijani | 35 | Hungarian   | 65 | Portuguese  |
| 6  | Bahraini    | 36 | Icelander   | 66 | Qatari      |
| 7  | Belgian     | 37 | Indian      | 67 | Romanian    |
| 8  | Bolivian    | 38 | Indonesian  | 68 | Russian     |

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

|    |             |    |                |    |               |
|----|-------------|----|----------------|----|---------------|
| 11 | Bulgarian   | 41 | Italian        | 71 | Scottish      |
| 12 | Cameroonian | 42 | Jamaican       | 72 | Serbian       |
| 13 | Canadian    | 43 | Japanese       | 73 | Singaporean   |
| 14 | Chilean     | 44 | Jordanian      | 74 | Slovakian     |
| 15 | Chinese     | 45 | Kuwaiti        | 75 | Slovenian     |
| 16 | Colombian   | 46 | Latvian        | 76 | South Korean  |
| 17 | Costa Rican | 47 | Lebanese       | 77 | South African |
| 18 | Croatian    | 48 | Lithuanian     | 78 | Spanish       |
| 19 | Cypriot     | 49 | Luxembourger   | 79 | Swedish       |
| 20 | Czech       | 50 | Malaysian      | 80 | Swiss         |
| 21 | Danish      | 51 | Maltese        | 81 | Thai          |
| 22 | Dutch       | 52 | Mexican        | 82 | Turkish       |
| 23 | Ecuadorian  | 53 | Monegasque     | 83 | Uruguayan     |
| 24 | English     | 54 | New Zealander  | 84 | Ukrainian     |
| 25 | Emirian     | 55 | Nicaraguan     | 85 | Venezuelan    |
| 26 | Estonian    | 56 | North Korean   | 86 | Welsh         |
| 27 | Finnish     | 57 | Northern Irish | 87 | Barbadian     |
| 28 | French      | 58 | Norwegian      | 88 | Vietnamese    |
| 29 | German      | 59 | Omani          |    |               |
| 30 | Ghanaian    | 60 | Pakistani      |    |               |

## Surface types

These types are from physics data and show what type of contact each wheel is experiencing.

| ID | Surface      |
|----|--------------|
| 0  | Tarmac       |
| 1  | Rumble strip |
| 2  | Concrete     |
| 3  | Rock         |
| 4  | Gravel       |

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

|    |             |
|----|-------------|
| 7  | Grass       |
| 8  | Water       |
| 9  | Cobblestone |
| 10 | Metal       |
| 11 | Ridged      |

## Button flags

These flags are used in the telemetry packet to determine if any buttons are being held on the controlling device. If the value below logical ANDed with the button status is set then the corresponding button is being held.

| Bit Flag | Button            |
|----------|-------------------|
| 0x0001   | Cross or A        |
| 0x0002   | Triangle or Y     |
| 0x0004   | Circle or B       |
| 0x0008   | Square or X       |
| 0x0010   | D-pad Left        |
| 0x0020   | D-pad Right       |
| 0x0040   | D-pad Up          |
| 0x0080   | D-pad Down        |
| 0x0100   | Options or Menu   |
| 0x0200   | L1 or LB          |
| 0x0400   | R1 or RB          |
| 0x0800   | L2 or LT          |
| 0x1000   | R2 or RT          |
| 0x2000   | Left Stick Click  |
| 0x4000   | Right Stick Click |

## Penalty types

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

|    |  |
|----|--|
| 1  | Stop Go  |
| 2  | Grid penalty                                     |
| 3  | Penalty reminder                                 |
| 4  | Time penalty                                     |
| 5  | Warning  |
| 6  | Disqualified                                     |
| 7  | Removed from formation lap                       |
| 8  | Parked too long timer                            |
| 9  | Tyre regulations                                 |
| 10 | This lap invalidated                             |
| 11 | This and next lap invalidated                    |
| 12 | This lap invalidated without reason              |
| 13 | This and next lap invalidated without reason     |
| 14 | This and previous lap invalidated                |
| 15 | This and previous lap invalidated without reason |
| 16 | Retired  |
| 17 | Black flag timer                                 |

## Infringement types

| ID | Infringement meaning                            |
|----|---|
| 0  | Blocking by slow driving                        |
| 1  | Blocking by wrong way driving                   |
| 2  | Reversing off the start line                    |
| 3  | Big Collision                                   |
| 4  | Small Collision                                 |
| 5  | Collision failed to hand back position single   |
| 6  | Collision failed to hand back position multiple |
| 7  | Corner cutting gained time                      |
| 8  | Corner cutting overtake single                  |

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

|    |   |
|----|---|
| 11 | Ignoring blue flags                             |
| 12 | Ignoring yellow flags                           |
| 13 | Ignoring drive through                          |
| 14 | Too many drive throughs                         |
| 15 | Drive through reminder serve within n laps      |
| 16 | Drive through reminder serve this lap           |
| 17 | Pit lane speeding                               |
| 18 | Parked for too long                             |
| 19 | Ignoring tyre regulations                       |
| 20 | Too many penalties                              |
| 21 | Multiple warnings                               |
| 22 | Approaching disqualification                    |
| 23 | Tyre regulations select single                  |
| 24 | Tyre regulations select multiple                |
| 25 | Lap invalidated corner cutting                  |
| 26 | Lap invalidated running wide                    |
| 27 | Corner cutting ran wide gained time minor       |
| 28 | Corner cutting ran wide gained time significant |
| 29 | Corner cutting ran wide gained time extreme     |
| 30 | Lap invalidated wall riding                     |
| 31 | Lap invalidated flashback used                  |
| 32 | Lap invalidated reset to track                  |
| 33 | Blocking the pitlane                            |
| 34 | Jump start                                      |
| 35 | Safety car to car collision                     |
| 36 | Safety car illegal overtake                     |
| 37 | Safety car exceeding allowed pace               |
| 38 | Virtual safety car exceeding allowed pace       |
| 39 | Formation lap below allowed speed               |

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)



F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!! [Read more...](#)



|    |                                 |
|----|---------------------------------|
| 42 | Safety car falling too far back |
| 43 | Black flag timer                |
| 44 | Unserved stop go penalty        |
| 45 | Unserved drive through penalty  |
| 46 | Engine component change         |
| 47 | Gearbox change                  |
| 48 | League grid penalty             |
| 49 | Retry penalty                   |
| 50 | Illegal time gain               |
| 51 | Mandatory pitstop               |



Drospy

Posted May 20



D

Members



5

34 posts

Hi,

is there any possibility to have "DELTA TIME" in UDP telemetry ?

Thank you

Walter

Hoo

Posted May 20



Codemasters Staff



+ 163

1,199 posts

On 5/20/2020 at 10:01 AM, Drospy said:



Hi,

is there any possibility to have "DELTA TIME" in UDP telemetry ?

Thank you

Walter

UP100

Posted May 20



Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

Moderator



+ 1,085  
2,868 posts

Is there any possibility to have DELTA TIME in UDP telemetry?

Thank you

Walter

Is there something you'd need delta time for? Maybe there's something else that could provide you the functionality that you require.

Pawel567

Posted May 20 (edited)



Members  
+ 20  
61 posts

On 5/20/2020 at 1:47 PM, UP100 said:



I would introduce for exceeding the delta instead of the penalty of passing through boxes a penalty of 5-10 seconds for a stop & go penalty that will take place in the box or will be added to you

I would have introduced for crossing the delta instead of a 5-10 sec penalty stop & go penalty which will take place in boxing or it will be added to you

Edited May 20 by Pawel567



Drospsy

Posted May 20



Members  
+ 5  
34 posts

On 5/20/2020 at 1:47 PM, UP100 said:



Is there something you'd need delta time for? Maybe there's something else that could provide you the functionality that you require.

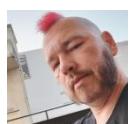
Now, calculate delta time between sector time and best time for sector. I think that this is a good info.

However, this doesn't work in Time Trial, because we don't have best time for sector



EnsiFerrum

Posted May 20



Members  
+ 84  
730 posts

I need the index of the current shown HUD page.

- CarSetupPanel -> Index = 0
- PitSetupPanel -> Index = 1
- CarDamagePanel -> Index = 2
- CarEnginePanel -> Index = 3
- CarTemperaturePanel -> Index = 4

Any possiblity to get this info?



cjorgens79

Posted May 21



Members  
+ 3

Just a minor point, I have been doing some api testing and have noticed that the two extra slots (21, 22) have randomish values in them for some fields that could make it a little confusing for some people working with the api. For example, the api is reporting 20 cars in the race (as expected), however slot 21 has semi valid x,z positional data, race position is 1 and lap number is 1. Slot 22 has no positional data, its race position is 0, but it is on lap 1. It would be nice if these slots could be cleared (zeroed out) properly so there is no randomish data in them to make it clearer for integrators when dealing with the telemetry data.

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

Oasis81



Members



4

49 posts

Posted May 22 (edited)

X

HI

same 2019 the data stops before the finish time in Q.

is it possible to extend the data sent until all the cars have finish the session?

second and most important thing for me:

the sectors: in the 2019 the sectors are erased after the lap, because at the new sector 1 in the new lap overwrite the sector 1 of the last lap.

this happen in every case, if the next lap is the pit entry or if it's a worst lap than the previous.

is it possible that the new sector overwrite the old sector only if the new lap is better than the previous?

Edited May 22 by Oasis81

Hoo



Codemasters Staff



+ 163

1,199 posts

Posted May 22

X

On 5/22/2020 at 6:15 PM, Oasis81 said:

X

HI

same 2019 the data stops before the finish time in Q.

is it possible to extend the data sent until all the cars have finish the session?

second and most important thing for me:

the sectors: in the 2019 the sectors are erased after the lap, because at the new sector 1 in the new lap overwrite the sector 1 of the last lap.

this happen in every case, if the next lap is the pit entry or if it's a worst lap than the previous.

is it possible that the new sector overwrite the old sector only if the new lap is better than the previous?

Hi @Oasis81 ,

In single player mode the session ends once the player has finished their session, so the UDP data simply reflects what the game is doing. If the game were to allow the session to continue and the player to spectate for the rest of the session (like it does in multiplayer) then this should just work in the UDP data. However, this is a bigger feature request that is beyond the scope of the UDP telemetry data. I can pass the suggestion back to the team if that is what you are proposing.

For the second point, the LapData packet currently keeps only the latest sector information. These times can obviously be stored by your app if needed. Are you suggesting that we should add a set of "best sector" times into the LapData packet?

Thanks,

Hoo.

Oasis81



Members



4

49 posts

Posted May 22 (edited)

X

On 5/22/2020 at 6:50 PM, Hoo said:

X

Hi @Oasis81 ,

In single player mode the session ends once the player has finished their session, so the UDP data simply reflects what the game is doing. If the game were to allow the session to continue and the player to spectate for the rest of the session (like it does in multiplayer) then this should just work in the UDP data. However, this is a bigger

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

into the LapData packet?

Thanks,  
Hoo.

Hello Hoo

thanks for the answer.

For the first point yes, the data stops when the player ends the Q. but usually there are some cars in the track that have no finished their laps, and this could be modify the classify. My suggestion is the choice to the player to stops immediatly the Q same as now, or to choose to continue to follow the Q in spectator mode same as multy, until all the cars are over the Q time, to have the UDP data also for this time left.

it will be also more realistic, because maybe in the Q I have the pole position, but when the Q finished at the next screen I'll see that i've lost it, but I couldn't followed this in real time.

for my view.

For the second point.

My app atm works same as this: every car has a personal list of record that stores every sector and every lap, order the list with the better lap and select the first record. , after this match this data with other list with other cars time and export the classify. this data can be stored only in real time mode,

ex:

car 1 | lap 3 | time 3 | sect 1.3 | sect 2.3 | sect 3.3

car 1 | lap 2 | time 2 | sect 1.2 | sect 2.2 | sect 3.2

car 1 | lap 1 | time 1 | sect 1.1 | sect 2.1 | sect 3.1

-----  
car 2 | lap 2 | time 2 | sect 1.2 | sect 2.2 | sect 3.2

car 2 | lap 1 | time 1 | sect 1.1 | sect 2.1 | sect 3.1

-----  
Class

car 1 | time 3 | sect 1.3 | sect 2.3 | sect 3.3

car 2 | time 2 | sect 1.2 | sect 2.2 | sect 3.2

ecc...

but this has 1 big problem:

1) if the player skip something, same as the pit exit or the return to the box or i FFW the time and other cars are on track and does their laps, , the data sent from the game about the sectors could be skipped, so i can't store it.

so, my suggestion is STORE the Table that the player seen on the monitor when is at the box , where is all the cars with their better time and the respective sectors.

because the game can store this data in some way, also if i skip something or FFW the time, but i can't.

I can suggest something like this:

m\_bestlap (there is)

m\_bestlapSector1

m\_bestlapSector2

m\_bestlapSector3

or only sect 1 and 2 could be fine.

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

thanks

regards

Edited May 22 by Oasis81



cjorgens79

Posted May 23



@Hoo

The LapData packet has some anomalous data in it. This is the current state of drivers 6 seconds into a 30 minute practice session.

Members  
3  
168 posts

Note that the Lap Distance and Total Distance values are large negatives. I would have expected these to be 0, as the session has just started and each driver is in the garage. The Pit Status also seems a bit random too, again everyone is in the garage however the pit status seems random between them. The very first LapData packet of the session looks exactly the same as well.

|                   |                             |                               |               |            |               |                  |
|-------------------|-----------------------------|-------------------------------|---------------|------------|---------------|------------------|
| D1 -> NIEVES      | sLapDist = -4152.5244140625 | sTotalDist = -4152.5244140625 | sRacePos = 6  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D2 -> FOREST      | sLapDist = -4238.380859375  | sTotalDist = -4238.380859375  | sRacePos = 2  | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |
| D3 -> MICHALSKI   | sLapDist = -4138.3212890625 | sTotalDist = -4138.3212890625 | sRacePos = 18 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D4 -> CALABRESI   | sLapDist = -4209.7451171875 | sTotalDist = -4209.7451171875 | sRacePos = 13 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D5 -> ATIYEH      | sLapDist = -4202.4130859375 | sTotalDist = -4202.4130859375 | sRacePos = 7  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D6 -> CLARKE      | sLapDist = -4195.6396484375 | sTotalDist = -4195.6396484375 | sRacePos = 10 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D7 -> LEVASSEUR   | sLapDist = -4252.5053710938 | sTotalDist = -4252.5053710938 | sRacePos = 12 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |
| D8 -> BELOUSOV    | sLapDist = -4159.8955078125 | sTotalDist = -4159.8955078125 | sRacePos = 11 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D9 -> CORREIA     | sLapDist = -4245.1743164063 | sTotalDist = -4245.1743164063 | sRacePos = 15 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |
| D10 -> IZUMI      | sLapDist = -4188.3076171875 | sTotalDist = -4188.3076171875 | sRacePos = 8  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D11 -> MURRAY     | sLapDist = -4259.23046875   | sTotalDist = -4259.23046875   | sRacePos = 3  | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |
| D12 -> SCHIFFER   | sLapDist = -4231.0493164063 | sTotalDist = -4231.0493164063 | sRacePos = 17 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |
| D13 -> KAUFMANN   | sLapDist = -4167.0986328125 | sTotalDist = -4167.0986328125 | sRacePos = 5  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D14 -> MORENO     | sLapDist = -4145.6923828125 | sTotalDist = -4145.6923828125 | sRacePos = 4  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D15 -> ROTH       | sLapDist = -4266.5610351563 | sTotalDist = -4266.5610351563 | sRacePos = 9  | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |
| D16 -> SAARI      | sLapDist = -4223.8271484375 | sTotalDist = -4223.8271484375 | sRacePos = 14 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D17 -> GILES      | sLapDist = -4302.1782226563 | sTotalDist = -4302.1782226563 | sRacePos = 19 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |
| D18 -> LAURSEN    | sLapDist = -4174.4619140625 | sTotalDist = -4174.4619140625 | sRacePos = 20 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D19 -> LETOURNEAU | sLapDist = -4216.4951171875 | sTotalDist = -4216.4951171875 | sRacePos = 1  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: 0 |
| D20 -> BARNES     | sLapDist = -4294.8837890625 | sTotalDist = -4294.8837890625 | sRacePos = 16 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: 0 |

cjorgens79

Posted May 23



Members  
3  
168 posts

On 5/22/2020 at 6:50 PM, Hoo said:

Hi @Oasis81 ,

In single player mode the session ends once the player has finished their session, so the UDP data simply reflects what the game is doing. If the game were to allow the session to continue and the player to spectate for the rest of the session (like it does in multiplayer) then this should just work in the UDP data. However, this is a bigger feature request that is beyond the scope of the UDP telemetry data. I can pass the suggestion back to the team if that is what you are proposing.

For the second point, the LapData packet currently keeps only the latest sector information. These times can obviously be stored by your app if needed. Are you suggesting that we should add a set of "best sector" times into the LapData packet?

Thanks,  
Hoo.

I think this could be useful as well for those single player sessions where time jumps occur by going back to garage or jumping to a flying lap. There is plenty of room in the packet to allow the addition of 2 more floats for

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!! [Read more...](#)

X

## LonelyRacer

Posted May 23



Members



60 posts

On 5/23/2020 at 3:15 AM, cjorgens79 said:



@Hoo

The LapData packet has some anomalous data in it. This is the current state of drivers 6 seconds into a 30 minute practice session.

Note that the Lap Distance and Total Distance values are large negatives. I would have expected these to be 0, as the session has just started and each driver is in the garage. The Pit Status also seems a bit random too, again everyone is in the garage however the pit status seems random between them. The very first LapData packet of the session looks exactly the same as well.

|                   |                             |                               |               |            |               |                |
|-------------------|-----------------------------|-------------------------------|---------------|------------|---------------|----------------|
| D1 -> NIEVES      | sLapDist = -4152.5244140625 | sTotalDist = -4152.5244140625 | sRacePos = 6  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D2 -> FOREST      | sLapDist = -4238.380859375  | sTotalDist = -4238.380859375  | sRacePos = 2  | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |
| D3 -> MICHALSKI   | sLapDist = -4138.3212890625 | sTotalDist = -4138.3212890625 | sRacePos = 18 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D4 -> CALABRESI   | sLapDist = -4209.7451171875 | sTotalDist = -4209.7451171875 | sRacePos = 13 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D5 -> ATIYEH      | sLapDist = -4202.4130859375 | sTotalDist = -4202.4130859375 | sRacePos = 7  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D6 -> CLARKE      | sLapDist = -4195.6396484375 | sTotalDist = -4195.6396484375 | sRacePos = 10 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D7 -> LEVASSEUR   | sLapDist = -4252.5053710938 | sTotalDist = -4252.5053710938 | sRacePos = 12 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |
| D8 -> BELOUSOV    | sLapDist = -4159.8955078125 | sTotalDist = -4159.8955078125 | sRacePos = 11 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D9 -> CORREIA     | sLapDist = -4245.1743164063 | sTotalDist = -4245.1743164063 | sRacePos = 15 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |
| D10 -> IZUMI      | sLapDist = -4188.3076171875 | sTotalDist = -4188.3076171875 | sRacePos = 8  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D11 -> MURRAY     | sLapDist = -4259.23046875   | sTotalDist = -4259.23046875   | sRacePos = 3  | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |
| D12 -> SCHIFFER   | sLapDist = -4231.0493164063 | sTotalDist = -4231.0493164063 | sRacePos = 17 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |
| D13 -> KAUFMANN   | sLapDist = -4167.0986328125 | sTotalDist = -4167.0986328125 | sRacePos = 5  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D14 -> MORENO     | sLapDist = -4145.6923828125 | sTotalDist = -4145.6923828125 | sRacePos = 4  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D15 -> ROTH       | sLapDist = -4266.5610351563 | sTotalDist = -4266.5610351563 | sRacePos = 9  | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |
| D16 -> SAARI      | sLapDist = -4223.8271484375 | sTotalDist = -4223.8271484375 | sRacePos = 14 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D17 -> GILES      | sLapDist = -4302.1782226563 | sTotalDist = -4302.1782226563 | sRacePos = 19 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |
| D18 -> LAURSEN    | sLapDist = -4174.4619140625 | sTotalDist = -4174.4619140625 | sRacePos = 20 | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D19 -> LETOURNEAU | sLapDist = -4216.4951171875 | sTotalDist = -4216.4951171875 | sRacePos = 1  | lapNum = 1 | mPitStatus: 0 | sDriverStatus: |
| D20 -> BARNES     | sLapDist = -4294.8837890625 | sTotalDist = -4294.8837890625 | sRacePos = 16 | lapNum = 1 | mPitStatus: 1 | sDriverStatus: |

The negative values mean, the car is that far of from the start/finish line. F1 2018 & F1 2019 this was the case with P & Q. So when you start your outlap, the values increase hitting 0 when your outlap finishes and the first real lap starts.

And the Trash in "unused" cars is also, what has been there at least in time trial and online races. If it doesn't get fixed/changed, then you have to massage the data yourself. That is what I have been doing with earlier games. I.e. by using the sResultStatus <= 1 to detect, if the driver is active and then modify inactive's values accordingly so they don't mess with the proper data.

The issue I see with the zeroing to "0" the values is that there is DriverID 0 for Carlos Sainz, so if they "0" all data, you still have the issue if the driver is "inactive" or if it is Sainz. I would prefer they change the Sainz driverId to something else. Would make many things easier.

I still need to do more driving and looking at the data to see, if there is anything odd/new going on this year. Thus far (some 3 hours of driving + some 15 hours or running save data feed on my tool), haven't noticed anything too critical.

## BernoAU

Posted May 24



Members



16 posts

It would be great if the LapData struct included a time gap to the car ahead.

All we need is the gap to car ahead as obviously the gap to car behind is the car ahead gap of the next position down. Also if we want to show the gap to the leader we can simply add up all the gaps ahead. We also already have the current lap number so can calculate if a car is a lap down.

So we could replicate the leaderboard with time gaps with just one extra float per driver in the struct.

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

Members  
3  
168 posts

The negative values mean, the car is that far from the start/finish line. F1 2018 & F1 2019 this was the case with P & Q. So when you start your outlap, the values increase hitting 0 when your outlap finishes and the first real lap starts.

And the Trash in "unused" cars is also, what has been there at least in time trial and online races. If it doesn't get fixed/changed, then you have to massage the data yourself. That is what I have been doing with earlier games. i.e. by using the sResultStatus <= 1 to detect, if the driver is active and then modify inactive's values accordingly so they don't mess with the proper data.

The issue I see with the zeroing to "0" the values is that there is DriverID 0 for Carlos Sainz, so if they "0" all data, you still have the issue if the driver is "inactive" or if it is Sainz. I would prefer they change the Sainz driverId to something else. Would make many things easier.

I still need to do more driving and looking at the data to see, if there is anything odd/new going on this year. Thus far (some 3 hours of driving + some 15 hours or running save data feed on my tool), haven't noticed anything too critical.

Thanks for the reply, after I posted it occurred to me that was probably the reason why the distances were negative. The pit status is a bug though as you have mentioned, it is easy enough to work around though.

Re the "zero" stuff, yeah i know what you mean about the conflicting ids, however the result status field's intention is to indicate whether or not the driver slot has valid data or not, if it is not a valid driver then any data in that driver's packet should be ignored. I found this flag after that original post, however i still think it is cleaner for the unused slots to be zeroed out.

DaveyGravy

Posted May 24

X

**D**

Members  
10  
53 posts

On 5/24/2020 at 2:45 AM, BernoAU said:

X

It would be great if the LapData struct included a time gap to the car ahead.

All we need is the gap to car ahead as obviously the gap to car behind is the car ahead gap of the next position down. Also if we want to show the gap to the leader we can simply add up all the gaps ahead. We also already have the current lap number so can calculate if a car is a lap down.

So we could replicate the leaderboard with time gaps with just one extra float per driver in the struct.

This would be super useful. I'm currently doing my own calculations to arrive at the time gaps which is OK, but direct data from the game would be much better and more accurate.

On a different note, it'd be nice to be able to know if DRS was going to be available soon, ie. the detection line has been crossed and will be available in the DRS zone. As far as I can tell this isn't covered.

Oasis81

Posted May 25 (edited)

X

**O**

Members  
4  
49 posts

a question

it's only mine problem or there is an error when you go on Q2 about the data?

the car with index 15 has CarPosition at 1, and my app generate errors because there are 2 cars with CarPosition at 1, the first and the 15

the problem is that the car is inactive because in the Q2 the Car 15 is out of qualify, so, same as other cars under it, the car position should be 0, not 1.

with f1 2019 I haven't this problem.

edit: after some checks i see that the first car out of range (15 in Q2) ... 11 in Q3 ... 21 in Race has always Car position at 1, and not at 0.

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X

```

    {
        LastLapPacketCarData[i].> [4] {Telemetry.LapData}
        LastLapPacketCarData[i].> [5] {Telemetry.LapData}
        LastLapPacketCarData[i].> [6] {Telemetry.LapData} data[i].Speed;
        LastLapPacketCarData[i].> [7] {Telemetry.LapData}
        LastLapPacketCarData[i].> [8] {Telemetry.LapData}
        LastLapPacketCarData[i].> [9] {Telemetry.LapData}
        LastLapPacketCarData[i].> [10] {Telemetry.LapData}
        LastLapPacketCarData[i].> [11] {Telemetry.LapData}
        LastLapPacketCarData[i].> [12] {Telemetry.LapData}
        LastLapPacketCarData[i].> [13] {Telemetry.LapData}
        LastLapPacketCarData[i].> [14] {Telemetry.LapData}
        LastLapPacketCarData[i].> [15] {Telemetry.LapData}
    }
}

private void _f1Manager_LapPacketReceivedEventArgs<PacketType>(PacketType e)
{
    InitializeCarGrid(e.Packet);
    if (sess == "Race")
    {
        CalculateCarSectors(e.Packet);
        for (int i = 0; i < 16; i++)
        {
            if (LastLapPacketCarData[i] == null)
            {
                continue;
            }
            if (packet.LapData[i] == null)
            {
                continue;
            }
        }
    }
}

```

|                   |          |
|-------------------|----------|
| BestLapTime       | 0        |
| CarPosition       | 1        |
| CurrentLapInvalid | 0        |
| CurrentLapNum     | 1        |
| CurrentLapTime    | 0        |
| DriverStatus      | InLap    |
| GridPosition      | 0        |
| LapDistance       | 5810.282 |
| LastLapTime       | 0        |
| Penalties         | 0        |
| PitStatus         | Pitting  |
| ResultStatus      | Invalid  |
| SafetyCarDelta    | 0        |
| Sector            | Sector3  |
| Sector1Time       | 0        |

Edited May 25 by Oasis81

cjorgens79

Posted May 26



Members  
3  
168 posts

On 5/25/2020 at 8:48 PM, Oasis81 said:

a question

it's only mine problem or there is an error when you go on Q2 about the data?

the car with index 15 has CarPosition at 1, and my app generate errors because there are 2 cars with CarPosition at 1., the first and the 15

the problem is that the car is inactive because in the Q2 the Car 15 is out of qualify, so, same as other cars under it, the car position should be 0, not 1.

with f1 2019 I haven't this problem.

edit: after some checks i see that the first car out of range (15 in Q2) ... 11 in Q3 ... 21 in Race has always Car position at 1, and not at 0.

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!! [Read more...](#)

```

    LastLapPacketCarData[i].> [6] {Telemetry.LapData} >ata[i].Speed;
    }
    CalculateCarDeltas();
}

1 riferimento
private void _f1Manager_LapPacketReceived(object sender, LapPacketReceivedEventArgs<Packet> e)
{
    InitializeCarGrid(e.Packet);
    if (sess == "Race")
    {
        CalculateCarSectors(e.Packet);
        for (int i = 0; i < 15; i++)
        {
            if (LastLapPacketCarData[i].CarPosition == 0)
            {
                continue;
            }
            if (packet.LapDistance > LastLapPacketCarData[i].LapDistance)
            {
                continue;
            }
            if (packet.CarPosition != LastLapPacketCarData[i].CarPosition)
            {
                continue;
            }
            if (packet.LapTime < LastLapPacketCarData[i].LastLapTime)
            {
                LastLapPacketCarData[i].LastLapTime = packet.LapTime;
                LastLapPacketCarData[i].CarPosition = packet.CarPosition;
                LastLapPacketCarData[i].DriverStatus = packet.DriverStatus;
                LastLapPacketCarData[i].GridPosition = packet.GridPosition;
                LastLapPacketCarData[i].LapDistance = packet.LapDistance;
                LastLapPacketCarData[i].Penalties = packet.Penalties;
                LastLapPacketCarData[i].PitStatus = packet.PitStatus;
                LastLapPacketCarData[i].ResultStatus = packet.ResultStatus;
                LastLapPacketCarData[i].SafetyCarDelta = packet.SafetyCarDelta;
                LastLapPacketCarData[i].Sector = packet.Sector;
                LastLapPacketCarData[i].Sector1Time = packet.Sector1Time;
            }
        }
    }
}

```

Check the `m_resultStatus` field, its intention is to indicate whether or not the driver slot has valid data or not, if it is not a valid driver then any data in that driver's packet should be ignored. So basically ignore any drivers who's `m_resultStatus` value is less than 2.

Oasis81

Posted May 26



Members  
4  
49 posts

Hi

Yes I must use that field now, the problem is that I must rewrite more and more part of the app because the packet comes with all the data, and without that carposition for invalid cars it was more easy to do, without filtering before.

Hoo

Posted May 26



Codemasters Staff



+ 163  
1,199 posts

Thanks for all of the feedback so far everyone. We're looking into some of these issues now.

In terms of delta times, we're not considering adding this in at the moment as the data should exist for everyone to do what they need. As stated above, delta time has so many different approaches and uses that it isn't something that we can add in to accommodate everyone's own use case. For example, there have been requests for the gap to the car in front, pitting delta, realtime delta used in Time Trial. Even something as simple as gap to the car in front comes with lots of design interpretation to make it work correctly:

If we simply work out the gap time gap between each car passing the specified point in the circuit, how do we interpolate between differences in track position reported by the car as these don't usually align? What happens when cars enter the pit lane or go off track? Do we consider using predictive deltas, or based them retrospectively on cars passing the same point in the track? What happens if you are not racing the car in front due to lapping them or alternative pitting strategies? What happens if cars are on in-laps or out-laps? ... and so on.

The game already uses a few different approaches to delta times and even more have been suggested by the community. As each one is a subjective design choice, we prefer to just provide the lap and time data and let you all choose how to use this.

We'll provide an update on the various changes and bug fixes ahead of the next beta phase. Thanks.



Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X



Members  
8  
60 posts

Just wanted to share this, as I see so many people asking about the Delta. And I know there are few ways to do this.

I used to calculate the delta as an estimate based on average speed and distances between the cars. Wasn't very good.

I know that some people store only leaders time and position and calculate the delta from that. This is pretty easy way to get the deltas.

My solution

This is what I have done in my tool: <https://www.racedepartment.com/downloads/telemetry-application.27456/>

The users can set the accuracy, basically the distance between detection points. Range is from 1m to 500m.

At the start of the race, I create an empty matrix, where each cell will contain <driverId, totalTime>. i.e. DeltaItem[columns][cars].

The width of the matrix (i.e. columns of the matrix) is: totalLapsInRace \* trackLength / accuracyDistance. For 5 lap race in 5k track with 100m accuracy the matrix width is 250 columns. With 20 cars, it means 5000 cells, each containing an int and float, so approx 50kb of memory. For a 70lap race in 5k track with 1m accuracy, memory used is about 70mb. For 10 hour endurance with 60 cars (in ACC), this would be around .3gb, but there e.g. 200m accuracy is more than sufficient, which takes about 1.5mb to store the whole 10h race.

At the start, the first column contains the situation at the start of the race. (totalDistanceDriven <= 0).

The index for each column after start is easily calculated with floor(totalDistanceDriven / accuracyDistance) + 1. So if accuracy is 50, you have driven 225, then the column is 5.

As the race commences and cars pass to the new "column", I store the leader to the top of the column and cars behind to lower positions, i.e. each column then contains from the top down the track positions at that point during the race. All cells in that column contain the driverId + totalTime at that point.

Note, all times use the totalTime, you can get that .e.g from the header's sessionTime.

So then when I calculate delta to the leader for certain car, I take their totalDistanceDriven, find the column. In that column I take their cell (based on driverId), then deduct from the total time the leaders total time (at row 0). This is the delta to the leader. E.g. leader entered column 6 at 17.1 and you entered there at 17.8, so delta is .7.

Delta to the car in front you get by finding the cars column (with totalDistanceDriven) + row (based on driverId), then calculating the totalTime diff for the row and row - 1. So you entered at column 6, your total time entering the range was 17.8. The car in front (your row - 1) entered the column at 17.6, so the delta to front is .2.

Delta to the car behind is similar, you just find the totalDistanceDriven for car behind, their correct column and then calculate the time difference to your cell on that column. The car behind is at column 5. Your time entering that range was 17.1 and the car behind entered at 17.5, so the delta to back is .4

Getting the data is pretty fast, as calls are direct array calls with index. The only small exception is to find the row, you compare with, which needs extra loop to find the right row with corresponding driverId, but in average it is MAX\_CARS/2, so with F1 series about 10 extra calls per search. But if you really need to optimize that too, you could store for the columns a hashmap<driverId, index\_in\_that\_column> to reduce this search also to a single call, but at a cost of memory (columns \* hashmap size)

One optimization one could do, is to store the data at the "game" order, i.e car at index 0 is stored at row 0 and so on. With F1 series this should be ok, as the order stays same during the race, but there are other games, where the "game order" changes, as people join/drop. This optimization would take away the driverId search mentioned above.

Other optimization for a dash solution would be to store only 4 rows, 1 for leader, 1 for car in front, 1 for the player and 1 for the car behind. Then you would always have leader at row 0, person in front (at that point) at row 1, the player in row 2 and the car behind in row 3. So the "id checking" would go away, but you would still know, who is in front/behind, if you save the driverId on the cells. And you would have access to the delta history to leader for the player. For 70lap on 6k track with 1m accuracy takes about 17mb.

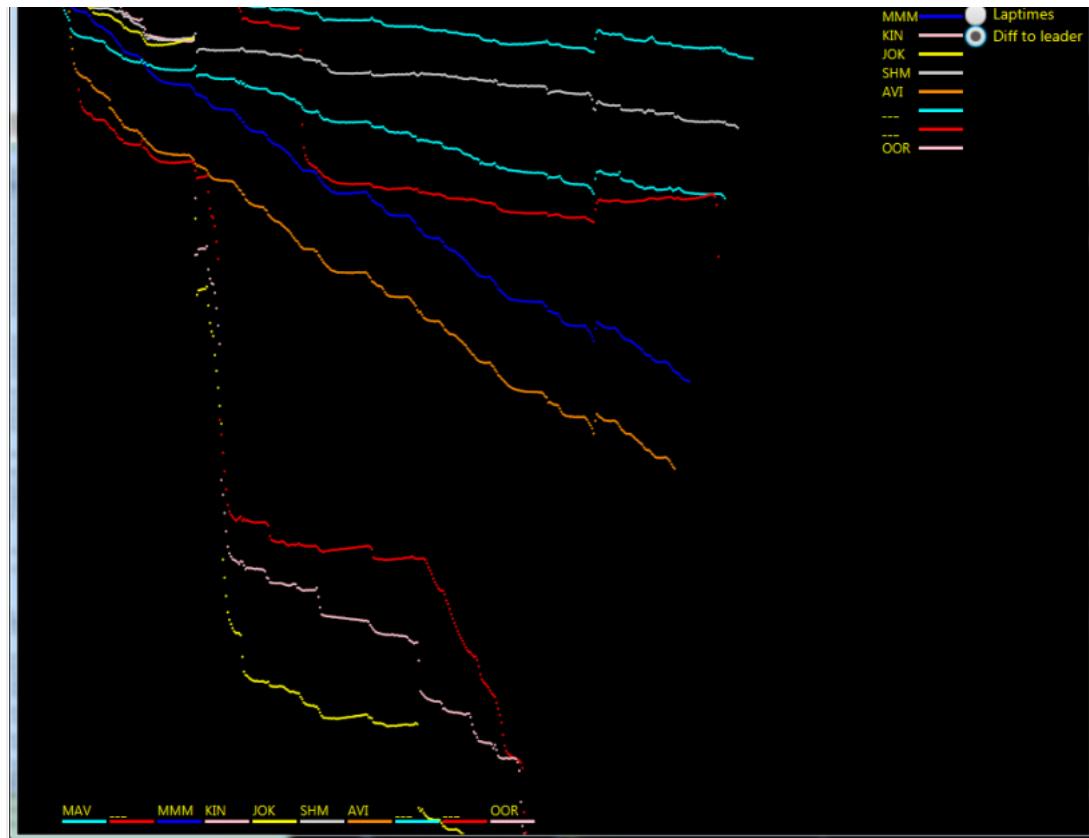
With this implementation, during the race and/or at the end of the race, I can produce a chart like this.

Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS [Read more...](#)

X

F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!! [Read more...](#)

X



Cheers.

Drospy

Posted May 27



On 5/26/2020 at 10:23 AM, Hoo said:

Thanks for all of the feedback so far everyone. We're looking into some of these issues now.

In terms of delta times, we're not considering adding this in at the moment as the data should exist for everyone to do what they need. As stated above, delta time has so many different approaches and uses that it isn't something that we can add in to accommodate everyone's own use case. For example, there have been requests for the gap to the car in front, pitting delta, realtime delta used in Time Trial. Even something as simple as gap to the car in front comes with lots of design interpretation to make it work correctly:

If we simply work out the gap time gap between each car passing the specified point in the circuit, how do we interpolate between differences in track position reported by the car as these don't usually align? What happens when cars enter the pit lane or go off track? Do we consider using predictive deltas, or based them retrospectively on cars passing the same point in the track? What happens if you are not racing the car in front due to lapping them or alternative pitting strategies? What happens if cars are on in-laps or out-laps? ... and so on.

The game already uses a few different approaches to delta times and even more have been suggested by the community. As each one is a subjective design choice, we prefer to just provide the lap and time data and let you all choose how to use this.

We'll provide an update on the various changes and bug fixes ahead of the next beta phase. Thanks.

Ok, all right, just a request:

In Time Trial there isn't data of bestlap, only best lap time, I ask to add SectorTime data of best lap.

Thank you for your support

[Bug Reporting - PLEASE FOLLOW RULES AND COMPLETE REPORTS](#) [Read more...](#)

X

[F1 2020 | PITCOINS NOW ASSIGNED - ONLINE SERVICES BACK ONLINE| READ ME!!!](#) [Read more...](#)

X

1 2 3 4 5 6 [NEXT](#) >> Page 1 of 1

## Create an account or sign in to comment

You need to be a member in order to leave a comment

### CREATE AN ACCOUNT

Sign up for a new account in our community. It's easy!

[Register a new account](#)

### SIGN IN

Already have an account? Sign in here.

[Sign In Now](#)

 [GO TO TOPIC LISTING](#)  
[Technical Assistance](#)



 [Home](#) > [F1 Games](#) > [F1® 2020 Game Forum](#) > [Technical Assistance](#) >

 [All Activity](#)

© 2018 The Codemasters Software Company Limited (“Codemasters”). All rights reserved.  
“Codemasters”® and the Codemasters logo® are registered trademarks owned by Codemasters. All Rights Reserved. All other trademarks or copyrights are the property of their respective owners and are used under license. Developed and published by Codemasters.

[CONTACT US](#)  
[TERMS & CONDITIONS](#)  
[PRIVACY POLICY](#)

Powered by Invision Community