**Application Protocol Data Unit (APDU)**

In the context of [smart cards](https://www.cardlogix.com/glossary/smart-card-definition/), an Application Protocol Data Unit (APDU) is the unit of communication between a [smart card reader](https://www.cardlogix.com/product-category/card-readers/) and a [smart card](https://www.cardlogix.com/product-category/smart-cards/). The structure of the APDU is defined by [ISO](https://www.cardlogix.com/glossary/iso/)/[IEC](https://www.cardlogix.com/glossary/iec-international-electrotechnical-commission/) 7816-4, which specifies organization, security, and commands for interchange.

An APDU consists of either a command message or a response message, which are sent from the interface device to the card or vice versa. A command-response pair is formed by sending a command, processing it in the receiving entity, and sending back a response. A specific response corresponds to a specific command.

The structure of a Command APDU (C-APDU) includes a required header (CLA INS P1 P2) and an optional body ([Lc field] [Data field] [Le field]). The CLA [byte](https://www.cardlogix.com/glossary/byte/) defines an application-specific class of instructions, and the INS byte defines a specific instruction within that class. The P1 and P2 fields can further qualify the instruction and provide input data. The Lc field defines the number of data [bytes](https://www.cardlogix.com/glossary/byte/) in the Data field, which can contain up to 255 bytes of data. The Le field defines the maximum number of bytes in the data field of the Response APDU (R-APDU).

The structure of a Response APDU (R-APDU) includes an optional body and a mandatory trailer. The Data field contains the response data, which can be up to 255 bytes and is returned by the [applet](https://www.cardlogix.com/glossary/java-applet/). The fields SW1 and SW2 provide feedback about the execution of the C-APDU, with the status word 0x9000 representing successful execution of the command.

**APDU message structure**

A step in an application protocol consists of sending a command, processing it in the receiving entity and sending back the response. Therefore a specific response corresponds to a specific command, referred to as a command-response pair.

An application protocol data unit (APDU) contains either a command message or a response message, sent from the interface device to the card or conversely. In a command-response pair, the command message and the response message may contain data, thus inducing four cases which are summarized in the table below.

**Command APDU**

A C-APDU consists of a required header (CLA INS P1 P2) and an optional body ([Lc field] [Data field] [Le field]). The CLA byte defines an application-specific class of instructions. According to the ISO7816 standard, byte values between 0xB0 and CF can be used. The INS byte defines a specific instruction within the class of instructions defined by the CLA byte. For valid CLA values, the application developer can define his own application specific instructions. The P1 and P2 fields can be used to further qualify the instruction and to provide input data respectively. The other fields are optional: the Lc field defines the number of data bytes in the data field; the Data field can contain up to 255 bytes of data; and the Le field defines the maximum number of bytes in the data field of the R-APDU.

**Response APDU**

An R-APDU consists of an optional body and mandatory trailer. The Data field contains the response data, maximum 255 bytes, returned by the applet. The fields SW1 and SW2 provide feedback about the execution of the C-APDU. Several status words are predefined in the ISO7816 standard. The status word 0x9000 represents successful execution of the command.

[**Java Card**](https://www.cardlogix.com/product-category/smart-cards/java-card/)

In [Java Card](https://www.cardlogix.com/glossary/java-card-smart/) programming, the applet receives the APDU instance to process from the [Java](https://www.cardlogix.com/glossary/java/) Card runtime environment in the [Applet](https://www.cardlogix.com/glossary/java-applet/). process (APDU) method. The APDU message is structured according to the ISO/IEC 7816-4 standard, which defines the organization, security, and commands for interchange between a [smart card](https://www.cardlogix.com/glossary/smart-card-definition/) and a card accepting device.

The APDU message structure consists of a header (CLA, INS, P1, P2, P3) and an optional body ([Lc field] [Data field] [Le field]) for command APDU, and an optional body and mandatory trailer (SW1, SW2) for response APDU. The APDU message may contain data, which is transferred as bytes.

The Java Card runtime environment provides an APDU class, which is used to manage the [APDU buffer](https://www.cardlogix.com/glossary/apdu-buffer/) and provides methods to read and write data to and from the APDU buffer. The APDU class [API](https://www.cardlogix.com/glossary/smart-card-api-application-programming-interface/) is designed to be transport protocol independent, which means that [applets](https://www.cardlogix.com/glossary/java-card-applet/) can use the same APDU methods regardless of the underlying protocol in use (e.g. [T=0](https://www.cardlogix.com/glossary/t0-protocol-smart-card/) or T=1 as defined in ISO 7816-3).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Command APDU** | | | | | | |
| **Header (required)** | | | | **Body (optional)** | | |
| CLA | INS | P1 | P2 | Lc | Data Field | Le |

|  |  |  |
| --- | --- | --- |
| **Command APDU** | | |
| **Field name** | **Length (bytes)** | **Description** |
| CLA | 1 | Instruction class – indicates the type of command, e.g. interindustry or proprietary |
| INS | 1 | Instruction code – indicates the specific command, e.g. “write data” |
| P1-P2 | 2 | Instruction parameters for the command, e.g. offset into file at which to write the data |
| Lc | 0, 1 or 3 | Encodes the number (Nc) of bytes of command data to follow  0 bytes denotes Nc=0 1 byte with a value from 1 to 255 denotes Nc with the same length 3 bytes, the first of which must be 0, denotes Nc in the range 1 to 65 535 (all three bytes may not be zero) |
| Command data | Nc | Nc bytes of data |
| Le | 0, 1, 2 or 3 | Encodes the maximum number (Ne) of response bytes expected  0 bytes denotes Ne=0 1 byte in the range 1 to 255 denotes that value of Ne, or 0 denotes Ne=256 2 bytes (if extended Lc was present in the command) in the range 1 to 65 535 denotes Ne of that value, or two zero bytes denotes 65 536 3 bytes (if Lc was not present in the command), the first of which must be 0, denote Ne in the same way as two-byte Le |
| **Response APDU** | | |
| Response data | Nr (at most Ne) | Response data |
| SW1-SW2 (Response trailer) | 2 | Command processing status, e.g. 90 00 (hexadecimal) indicates success |
| SW1-SW2 | 2 | Command processing status, e.g. 6A 82 (hexadecimal) indicates data object not found |
| SW1-SW2 | 2 | Command processing status, e.g. 69 82 (hexadecimal) indicates security status not satisfied |

List of most common APDU commands:

1. SELECT FILE (0x00A4)
2. READ BINARY (0x00B0)
3. UPDATE BINARY (0x00D6)
4. READ RECORDS (0x00B2)
5. UPDATE RECORDS (0x00DC)
6. APPEND RECORD (0x00E2)
7. VERIFY (0x0020)
8. CHANGE REFERENCE DATA (0x0024)
9. RESET RETRY COUNTER (0x002C)
10. GET CHALLENGE (0x0084)
11. INTERNAL AUTHENTICATE (0x0088)
12. EXTERNAL AUTHENTICATE (0x0082)
13. TERMINATE CARD USAGE (0x826E)
14. ERASE BINARY (0x00DA)
15. ERASE RECORDS (0x00DC)

Note that this is not an exhaustive list and there are many other APDU commands defined in the ISO/IEC 7816-4 standard.