It is often difficult to find a single "complete" list because strictly speaking, **ISO 8859-1 does not define the control characters itself.** It only defines the printable graphic characters and *reserves* the empty spots (0x00–0x1F and 0x80–0x9F) for control codes.[1]

In practice, however, these spots are almost always filled by the **C0** and **C1** control sets defined in ISO 6429.[2]

Below is the complete breakdown, specifically detailing the "hidden" control characters you were looking for.

Note: The first 255 characters from the ISO8859-1 code page are exactly the same as the first 255 characters from UTF-8. Characters on positions greater or equal than 128, will map one after another starting with 49792 which in hex is C280. The characters can easily be found by the code points as they follow the sequence (i.e. characters are encoded from byte 00 to C3BF with a gap while codepoints go from 0000 to 00FF uninterrupted).

1. C0 Control Characters (0x00 – 0x1F)

These are the standard **ASCII control characters**.

Hex	Dec	Name	Description
00	0	**NUL**	Null
01	1	**SOH**	Start of Heading
02	2	**STX**	Start of Text
03	3	**ETX**	End of Text
04	4	**EOT**	End of Transmission
05	5	**ENQ**	Enquiry
06	6	**ACK**	Acknowledge
07	7	**BEL**	Bell
08	8	**BS**	Backspace

09	9	**HT**	Horizontal Tab
0A	10	**LF**	Line Feed
0B	11	**VT**	Vertical Tab
0C	12	**FF**	Form Feed
0D	13	**CR**	Carriage Return
0E	14	**SO**	Shift Out
0F	15	**SI**	Shift In
10	16	**DLE**	Data Link Escape
11	17	**DC1**	Device Control 1 (XON)
12	18	**DC2**	Device Control 2
13	19	**DC3**	Device Control 3 (XOFF)
14	20	**DC4**	Device Control 4
15	21	**NAK**	Negative Acknowledge
16	22	**SYN**	Synchronous Idle
17	23	**ETB**	End of Transmission Block
18	24	**CAN**	Cancel
19	25	**EM**	End of Medium

1A	26	**SUB**	Substitute
1B	27	**ESC**	Escape
1C	28	**FS**	File Separator
1D	29	**GS**	Group Separator
1E	30	**RS**	Record Separator
1F	31	**US**	Unit Separator

---

2. Standard Printable Characters (0x20 – 0x7E)

These are standard ASCII characters (A-Z, 0-9, punctuation).[3]

- **0x20**: SPACE[4][5][6]
- **0x21 – 0x7E**: ! " # $ % & ' ( ) * + , - . / 0-9 : ; < = > ? @ A-Z [\ ] ^ _ a-z { | } ~ `[7][8]

3. Delete Control Character (0x7F)[9][10]

Hex	Dec	Name	Description
7F[19][20]	127[21][22]	**DEL**[23][24]	Delete[25][26]

---

4. C1 Control Characters (0x80 – 0x9F)

This is the missing piece. Many web pages leave these blank or label them "undefined" because ISO 8859-1 technically leaves them empty. However, systems adhering to the ISO 2022 structure map these to the **C1 Control Set** (ISO 6429).

Hex	Dec	Name	Description
80	128	**PAD**	Padding Character

81	129	**HOP**	High Octet Preset
82	130	**BPH**	Break Permitted Here
83	131	**NBH**	No Break Here
84	132	**IND**	Index
85	133	**NEL**	Next Line
86	134	**SSA**	Start of Selected Area
87	135	**ESA**	End of Selected Area
88	136	**HTS**	Horizontal Tabulation Set
89	137	**HTJ**	Horizontal Tabulation with Justification
8A	138	**VTS**	Vertical Tabulation Set
8B	139	**PLD**	Partial Line Forward (Down)
8C	140	**PLU**	Partial Line Backward (Up)
8D	141	**RI**	Reverse Line Feed (Index)
8E	142	**SS2**	Single Shift Two

8F	143	**SS3**	Single Shift Three
90	144	**DCS**	Device Control String
91	145	**PU1**	Private Use One
92	146	**PU2**	Private Use Two
93	147	**STS**	Set Transmit State
94	148	**CCH**	Cancel Character
95	149	**MW**	Message Waiting
96	150	**SPA**	Start of Guarded Area
97	151	**EPA**	End of Guarded Area
98	152	**SOS**	Start of String
99	153	**SGCI**	Single Graphic Character Introducer
9A	154	**SCI**	Single Character Introducer
9B	155	**CSI**	Control Sequence Introducer
9C	156	**ST**	String Terminator
9D	157	**OSC**	Operating System Command

9E	158	**PM**	Privacy Message
9F	159	**APC**	Application Program Command

---

5. ISO 8859-1 Extended Printable (0xA0 – 0xFF)

These are the Western European specific characters (accented letters, currency, copyright, etc.).

- **0xA0**: No-Break Space (NBSP)
- **0xA1 – 0xFF**: ¡ ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿ À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ð Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ

---

6. Is ISO 2022 the latest standard?

No. While ISO/IEC 2022 (ECMA-35) is still a valid standard on the books (the latest version is from 1994), it is **not** the modern standard for character encoding.

- **The Modern Standard:** The world has moved to **ISO/IEC 10646 (Unicode)**. Unicode abandoned the complex "switching" mechanisms of ISO 2022 in favor of a single, universal character set that assigns a unique number to every character.
- **The Legacy Role:** ISO 2022 is the "parent" standard that defines how 7-bit and 8-bit character sets (like ISO 8859-1) work. If you are using ISO 8859-1 today, you are technically using an "instance" of the ISO 2022 structure.

7. How were control characters handled *before* ISO 2022?

Before ISO 2022 standardized the structure (in the early 1970s), control characters were largely **ad-hoc** or limited to 7-bit environments.

- **Teleprinters (Baudot Code):** In the pre-computer era (1930s-50s), 5-bit codes were used. They didn't have a "control set" in the modern sense. Instead, they had two specific "shift" characters: **FIGS** (Figures) and **LTRS** (Letters). You had to send a specific byte to switch the machine into "number mode" or "letter mode."
- **ASCII (The C0 Set):** When ASCII (7-bit) was finalized in the 1960s, it defined the first 32 bytes (0x00–0x1F) as control characters. This became known as the **C0 Set**.
- **The Chaos:** As computers moved to 8-bit bytes, vendors (like IBM with EBCDIC) started filling the extra space with whatever they wanted—sometimes more controls, sometimes graphics. There was no agreement on *where* in the 8-bit map extra control codes should live.

8. Why was the C1 Control Set from ISO 6429 chosen?

This is the key to your original question about the "hidden" characters in ISO 8859-1.

When the ISO committee designed the **ISO 8859** family (which includes 8859-1), they wanted it to be strictly compatible with the **ISO 2022** structural rules.

A. The ISO 2022 "Container"

ISO 2022 acts as a container or template. It states that a valid 8-bit character set must be split into four specific regions:

- **C0 (0x00–0x1F):** Reserved for standard ASCII controls.
- **G0 (0x20–0x7E):** The standard printable characters (ASCII).
- **C1 (0x80–0x9F): Reserved for "extended" control characters.**
- **G1 (0xA0–0xFF):** The extended printable characters (e.g., accented letters).

B. The ISO 6429 "Content"

ISO 2022 defined *where* the controls go (the C1 range), but it didn't define *what* they should do.

The committee chose **ISO 6429 (also known as ECMA-48)** to fill this gap because it was already the industry standard for controlling "intelligent" terminals (like the famous DEC VT100).

- **The Gap Filler:** ISO 6429 defined functions for things 7-bit ASCII couldn't handle, like moving the cursor up and down, bolding text, or changing colors.
- **The Result:** To remain compliant with the ISO 2022 structure, ISO 8859-1 was forced to leave the 0x80–0x9F range empty of printable graphics, implicitly reserving it for the ISO 6429 C1 control functions.

> **Summary:** ISO 2022 built the **house** (the structure requiring a C1 slot). ISO 6429 provided the **furniture** (the actual control functions). ISO 8859-1 lives in that house, which is why it can't put printable characters in the C1 room.

Sources

Here are the specific standards and technical documents that validate the information provided in the previous responses.

1. The "Hidden" C1 Control List (0x80–0x9F)

The specific list of control characters (PAD, HOP, BPH, etc.) that fill the 0x80–0x9F gap is defined in **ISO/IEC 6429** (also known as **ECMA-48**).

- **Source:** *Standard ECMA-48: Control Functions for Coded Character Sets* (5th Edition, 1991).
    - **Relevant Section:** Table 2 ("C1 Control Characters") lists the specific names and descriptions for bytes 08/00 (0x80) through 09/15 (0x9F).
- **Verification:** You can view the list in the ECMA-48 Standard PDF.

2. ISO 8859-1 Structure & Reserved Ranges

The rule that ISO 8859-1 must keep 0x00–0x1F and 0x80–0x9F empty of graphic characters comes from the structural definitions in **ISO/IEC 2022** and **ISO/IEC 8859-1** itself.

- **Source:** *ISO/IEC 8859-1:1998 Information technology — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1.*
    - **Relevant Fact:** The standard explicitly defines the "C0" and "C1" ranges as reserved for control functions and does not assign them graphic symbols.
- **Source:** *RFC 2130 (The Report of the IAB Character Set Workshop).*
    - **Relevant Fact:** This document explains the architecture where ISO 8859 series standards define the **G** (Graphic) sets but rely on ISO 6429 for the **C** (Control) sets.

3. The ISO 2022 "Container" Logic

The explanation regarding C0, C1, G0, and G1 regions is based on **ISO/IEC 2022** (also known as **ECMA-35**).

- **Source:** *Standard ECMA-35: Character Code Structure and Extension Techniques* (6th Edition, 1994).
    - **Relevant Section:** Section 8 ("Structure of 8-bit codes") details the matrix containing the C0, G0, C1, and G1 areas.
    - **Verification:** Available via ECMA International.

Standard	Common Name	Role in this Context
ISO/IEC 2022	ECMA-35	The **Structure**. Defines that an 8-bit set must have a "C1" control region at 0x80–0x9F.
ISO/IEC 6429	ECMA-48	The **Content**. Defines the actual list of control functions (PAD, IND, NEL, etc.) that go into that region.
ISO/IEC 8859-1	Latin-1	The **Instance**. An implementation of the

		structure that fills the graphic regions (G0, G1) but reserves the control regions.