


# WebAssembly Code Metadata Specification

**WebAssembly Community Group**

**Yuri Iozzelli (editor)**

**Nov 25, 2026**

## Contents

1 Introduction 1

2 Binary Format 2

3 Text Format 2

Index 3

## 1 Introduction

This document defines a generic mechanism for attaching arbitrary metadata to WebAssembly instructions. Additionally, it defines specific metadata formats using this mechanism.

Such metadata does not contribute to, or otherwise affect, the WebAssembly semantics, and may be ignored by an implementation.

However, it can provides useful information that implementations can make use of to improve user experience or take compilation hints.

### 1.1 Dependencies

This document is based on the WebAssembly core specification (https://webassembly.github.io/spec/core/), and makes use of terms and definitions described there. These uses always link to the corresponding definition in the core specification.

# 2 Binary Format

## 2.1 Code Metadata

A Code Metadata item is a piece of information logically attached to an instruction.

An item is associated with a format, which defines the item's payload.

All code metadata items of a format named $T$ are grouped under a custom section named *'metadata.code.T'*. The following parametrized grammar rules define the generic structure of a code metadata section of format $T$.

$$
\begin{array}{lcll}
\texttt{codemetadatasec}(\texttt{T}) & ::= & \texttt{section}_0(\texttt{codemetadata}(\texttt{T})) & \\
\texttt{codemetadata}(\texttt{T}) & ::= & n{:}\texttt{name} & (\text{if } n = \text{`}\texttt{metadata.code.T}\text{'}) \\
 & & \texttt{list}(\texttt{codemetadatafunc}(\texttt{T})) & \\
\texttt{codemetadatafunc}(\texttt{T}) & ::= & x{:}\texttt{funcidx}\ \texttt{list}(\texttt{codemetadataitem}(\texttt{T})) & \\
\texttt{codemetadataitem}(\texttt{T}) & ::= & \mathit{off}{:}\texttt{u32}\ \ \mathit{size}{:}\texttt{u32} & (\text{if } \mathit{size} = ||\texttt{T}||) \\
 & & \mathit{data}{:}\texttt{T} &
\end{array}
$$

Where $off$ is the byte offset of the attached instruction, relative to the beginning of the `func` declaration, and $data$ is a further payload, whose content depends on the format $T$.

`codemetadatafunc` entries must appear in order of increasing $x$, and duplicate id values are not allowed. `codemetadata` entries must appear in order of increasing $off$, and duplicate offset values are not allowed.

### Branch Hints

A Branch Hint is a code metadata item with format *branch_hint*.

It can only be attached to br_if and if instructions.

Its payload indicates whether the branch is likely or unlikely to be taken.

All branch hints for a module are contained in a single code metadata section with name *'metadata.code.branch_hint'*.

$$
\begin{array}{lcl}
\texttt{branchhintsec} & ::= & \texttt{codemetadatasec}(\texttt{branchhint}) \\
\texttt{branchhint} & ::= & \texttt{unlikely} \\
 & | & \texttt{likely} \\
\texttt{unlikely} & ::= & \texttt{0x00} \\
\texttt{likely} & ::= & \texttt{0x01}
\end{array}
$$

A value of `likely` means that the branch is likely to be taken, while a value of `unlikely` means the opposite. A branch with no hints is considered equally likely to be taken or not.

# 3 Text Format

## 3.1 Code Metadata

Code Metadata items appear in the text format as custom annotations, and are considered attached to the first instruction that follows them.

$$
\texttt{codemetadataannot}(\texttt{T}) \quad ::= \quad \text{`}(\texttt{@metadata.code.T}\text{'}\ \mathit{data}{:}\texttt{T}\ \text{`})\text{'}
$$

Where $T$ is the format name of the item, and *data* is a byte string containing the same payload as in the binary format.

# Index


## B

branch hint section, 2

## C

code metadata annotation, 2
code metadata section, 2