Module: tf.dtypes

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes#top_of_page)
* [Classes](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes#classes)
* [Functions](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes#functions)
* [Other Members](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes#other_members)

Public API for tf.dtypes namespace.

Classes

[class DType](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType): Represents the type of the elements in a Tensor.

Functions

[as\_dtype(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/as_dtype): Converts the given type\_value to a DType.

[cast(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/cast): Casts a tensor to a new type.

[complex(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/complex): Converts two real numbers to a complex number.

[saturate\_cast(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/saturate_cast): Performs a safe saturating cast of value to dtype.

Other Members

* int32
* int64
* int8
* qint16
* qint32
* qint8
* quint16
* quint8
* resource
* string
* uint16
* uint32
* uint64
* QUANTIZED\_DTYPES
* bfloat16
* bool
* complex128
* complex64
* double
* float16
* float32
* float64
* half
* int16
* uint8
* variant

# tf.dtypes.as\_dtype

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/as_dtype#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/as_dtype#aliases)

Converts the given type\_value to a DType.

### Aliases:

* tf.as\_dtype
* tf.compat.v1.as\_dtype
* tf.compat.v1.dtypes.as\_dtype
* tf.compat.v2.as\_dtype
* tf.compat.v2.dtypes.as\_dtype
* tf.dtypes.as\_dtype

tf.dtypes.as\_dtype(type\_value)

Defined in [python/framework/dtypes.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/framework/dtypes.py).

#### Args:

* **type\_value**: A value that can be converted to a [tf.DType](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType) object. This may currently be a [tf.DType](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType) object, a [DataType enum](https://www.tensorflow.org/code/tensorflow/core/framework/types.proto), a string type name, or a numpy.dtype.

#### Returns:

A DType corresponding to type\_value.

#### Raises:

* **TypeError**: If type\_value cannot be converted to a DType.

# tf.dtypes.cast

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/cast#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/cast#aliases)
* [Used in the guide:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/cast#used_in_the_guide)
* [Used in the tutorials:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/cast#used_in_the_tutorials)

Casts a tensor to a new type.

### Aliases:

* tf.cast
* tf.compat.v1.cast
* tf.compat.v1.dtypes.cast
* tf.compat.v2.cast
* tf.compat.v2.dtypes.cast
* tf.dtypes.cast

tf.dtypes.cast(  
    x,  
    dtype,  
    name=None  
)

Defined in [python/ops/math\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/math_ops.py).

### Used in the guide:

* [Convert Your Existing Code to TensorFlow 2.0](https://www.tensorflow.org/beta/guide/migration_guide)
* [Eager essentials](https://www.tensorflow.org/beta/guide/eager)
* [Keras: A quick overview](https://www.tensorflow.org/beta/guide/keras/overview)
* [Training and Evaluation with TensorFlow Keras](https://www.tensorflow.org/beta/guide/keras/training_and_evaluation)
* [tf.function and AutoGraph in TensorFlow 2.0](https://www.tensorflow.org/beta/guide/autograph)

### Used in the tutorials:

* [Distributed training with Keras](https://www.tensorflow.org/beta/tutorials/distribute/keras)
* [Image Captioning with Attention](https://www.tensorflow.org/beta/tutorials/text/image_captioning)
* [Load CSV with tf.data](https://www.tensorflow.org/beta/tutorials/load_data/csv)
* [Load images with tf.data](https://www.tensorflow.org/beta/tutorials/load_data/images)
* [Load text with tf.data](https://www.tensorflow.org/beta/tutorials/load_data/text)
* [Multi-worker Training with Estimator](https://www.tensorflow.org/beta/tutorials/distribute/multi_worker_with_estimator)
* [Multi-worker Training with Keras](https://www.tensorflow.org/beta/tutorials/distribute/multi_worker_with_keras)
* [Neural Machine Translation with Attention](https://www.tensorflow.org/beta/tutorials/text/nmt_with_attention)
* [Neural style transfer](https://www.tensorflow.org/beta/tutorials/generative/style_transfer)
* [Pix2Pix](https://www.tensorflow.org/beta/tutorials/generative/pix2pix)
* [Transfer Learning Using Pretrained ConvNets](https://www.tensorflow.org/beta/tutorials/images/transfer_learning)
* [Transformer model for language understanding](https://www.tensorflow.org/beta/tutorials/text/transformer)
* [Unicode strings](https://www.tensorflow.org/beta/tutorials/text/unicode)
* [tf.distribute.Strategy with training loops](https://www.tensorflow.org/beta/tutorials/distribute/training_loops)

The operation casts x (in case of Tensor) or x.values (in case of SparseTensor or IndexedSlices) to dtype.

#### For example:

x = tf.constant([1.8, 2.2], dtype=tf.float32)  
tf.cast(x, tf.int32)  # [1, 2], dtype=tf.int32

The operation supports data types (for x and dtype) of uint8, uint16, uint32, uint64, int8, int16, int32, int64, float16, float32, float64, complex64, complex128, bfloat16. In case of casting from complex types (complex64, complex128) to real types, only the real part of xis returned. In case of casting from real types to complex types (complex64, complex128), the imaginary part of the returned value is set to 0. The handling of complex types here matches the behavior of numpy.

#### Args:

* **x**: A Tensor or SparseTensor or IndexedSlices of numeric type. It could be uint8, uint16, uint32, uint64, int8, int16, int32, int64, float16, float32, float64, complex64, complex128, bfloat16.
* **dtype**: The destination type. The list of supported dtypes is the same as x.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor or SparseTensor or IndexedSlices with same shape as x and same type as dtype.

#### Raises:

* **TypeError**: If x cannot be cast to the dtype.

# tf.dtypes.complex

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/complex#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/complex#aliases)

Converts two real numbers to a complex number.

### Aliases:

* tf.compat.v1.complex
* tf.compat.v1.dtypes.complex
* tf.compat.v2.complex
* tf.compat.v2.dtypes.complex
* tf.complex
* tf.dtypes.complex

tf.dtypes.complex(  
    real,  
    imag,  
    name=None  
)

Defined in [python/ops/math\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/math_ops.py).

Given a tensor real representing the real part of a complex number, and a tensor imag representing the imaginary part of a complex number, this operation returns complex numbers elementwise of the form a+bj, where a represents the real part and b represents the imag part.

The input tensors real and imag must have the same shape.

#### For example:

real = tf.constant([2.25, 3.25])  
imag = tf.constant([4.75, 5.75])  
tf.complex(real, imag)  # [[2.25 + 4.75j], [3.25 + 5.75j]]

#### Args:

* **real**: A Tensor. Must be one of the following types: float32, float64.
* **imag**: A Tensor. Must have the same type as real.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor of type complex64 or complex128.

# tf.dtypes.DType

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType#top_of_page)
* [Class DType](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType#class_dtype)
  + [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType#aliases)
* [\_\_init\_\_](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType#__init__)
* [Properties](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType#properties)

## Class DType

Represents the type of the elements in a Tensor.

### Aliases:

* Class tf.DType
* Class tf.compat.v1.DType
* Class tf.compat.v1.dtypes.DType
* Class tf.compat.v2.DType
* Class tf.compat.v2.dtypes.DType
* Class tf.dtypes.DType

Defined in [python/framework/dtypes.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/framework/dtypes.py).

The following DType objects are defined:

* [tf.float16](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#float16): 16-bit half-precision floating-point.
* [tf.float32](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#float32): 32-bit single-precision floating-point.
* [tf.float64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#float64): 64-bit double-precision floating-point.
* [tf.bfloat16](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#bfloat16): 16-bit truncated floating-point.
* [tf.complex64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#complex64): 64-bit single-precision complex.
* [tf.complex128](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#complex128): 128-bit double-precision complex.
* [tf.int8](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int8): 8-bit signed integer.
* [tf.uint8](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#uint8): 8-bit unsigned integer.
* [tf.uint16](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#uint16): 16-bit unsigned integer.
* [tf.uint32](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#uint32): 32-bit unsigned integer.
* [tf.uint64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#uint64): 64-bit unsigned integer.
* [tf.int16](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int16): 16-bit signed integer.
* [tf.int32](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int32): 32-bit signed integer.
* [tf.int64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int64): 64-bit signed integer.
* [tf.bool](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#bool): Boolean.
* [tf.string](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#string): String.
* [tf.qint8](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#qint8): Quantized 8-bit signed integer.
* [tf.quint8](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#quint8): Quantized 8-bit unsigned integer.
* [tf.qint16](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#qint16): Quantized 16-bit signed integer.
* [tf.quint16](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#quint16): Quantized 16-bit unsigned integer.
* [tf.qint32](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#qint32): Quantized 32-bit signed integer.
* [tf.resource](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#resource): Handle to a mutable resource.
* [tf.variant](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#variant): Values of arbitrary types.

The tf.as\_dtype() function converts numpy types and string type names to a DType object.

## \_\_init\_\_

\_\_init\_\_(type\_enum)

Creates a new DataType.

NOTE(mrry): In normal circumstances, you should not need to construct a DataType object directly. Instead, use the tf.as\_dtype() function.

#### Args:

* **type\_enum**: A types\_pb2.DataType enum value.

#### Raises:

* **TypeError**: If type\_enum is not a value types\_pb2.DataType.

## Properties

### as\_datatype\_enum

Returns a types\_pb2.DataType enum value based on this DType.

### as\_numpy\_dtype

Returns a numpy.dtype based on this DType.

### base\_dtype

Returns a non-reference DType based on this DType.

### is\_bool

Returns whether this is a boolean data type

### is\_complex

Returns whether this is a complex floating point type.

### is\_floating

Returns whether this is a (non-quantized, real) floating point type.

### is\_integer

Returns whether this is a (non-quantized) integer type.

### is\_numpy\_compatible

### is\_quantized

Returns whether this is a quantized data type.

### is\_unsigned

Returns whether this type is unsigned.

Non-numeric, unordered, and quantized types are not considered unsigned, and this function returns False.

#### Returns:

Whether a DType is unsigned.

### limits

Return intensity limits, i.e.

(min, max) tuple, of the dtype. Args: clip\_negative : bool, optional If True, clip the negative range (i.e. return 0 for min intensity) even if the image dtype allows negative values. Returns min, max : tuple Lower and upper intensity limits.

### max

Returns the maximum representable value in this data type.

#### Raises:

* **TypeError**: if this is a non-numeric, unordered, or quantized type.

### min

Returns the minimum representable value in this data type.

#### Raises:

* **TypeError**: if this is a non-numeric, unordered, or quantized type.

### name

Returns the string name for this DType.

### real\_dtype

Returns the dtype correspond to this dtype's real part.

### size

## Methods

### \_\_eq\_\_

\_\_eq\_\_(other)

Returns True iff this DType refers to the same type as other.

### \_\_ne\_\_

\_\_ne\_\_(other)

Returns True iff self != other.

### is\_compatible\_with

is\_compatible\_with(other)

Returns True if the other DType will be converted to this DType.

The conversion rules are as follows:

DType(T)       .is\_compatible\_with(DType(T))        == True

#### Args:

* **other**: A DType (or object that may be converted to a DType).

#### Returns:

True if a Tensor of the other DType will be implicitly converted to this DType.

# tf.dtypes.saturate\_cast

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/saturate_cast#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/saturate_cast#aliases)

Performs a safe saturating cast of value to dtype.

### Aliases:

* tf.compat.v1.dtypes.saturate\_cast
* tf.compat.v1.saturate\_cast
* tf.compat.v2.dtypes.saturate\_cast
* tf.compat.v2.saturate\_cast
* tf.dtypes.saturate\_cast
* tf.saturate\_cast

tf.dtypes.saturate\_cast(  
    value,  
    dtype,  
    name=None  
)

Defined in [python/ops/math\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/math_ops.py).

This function casts the input to dtype without applying any scaling. If there is a danger that values would over or underflow in the cast, this op applies the appropriate clamping before the cast.

#### Args:

* **value**: A Tensor.
* **dtype**: The desired output DType.
* **name**: A name for the operation (optional).

#### Returns:

value safely cast to dtype.