Guava Release 19.0: Release Notes

- 19.0 was released on December 9, 2015.
- 19.0-rc3 was released on December 1, 2015.
- 19.0-rc2 was released on September 17, 2015.
- 19.0-rc1 was released on July 23, 2015.

(See [[ReleaseHistory]].)

API documentation: * guava * guava-testlib

Using Guava in your project

	Guava	**Guava (GWT)**
Maven Identifier	com.google.guava:guava:19.0	com.google.guava:guava-gwt:19.0
Jar	guava-19.0.jar	guava-gwt-19.0.jar
Javadoc	guava-19.0-javadoc.jar	guava-gwt-19.0-javadoc.jar
Sources	guava-19.0-sources.jar	guava-gwt-19.0-sources.jar

See [[UseGuavaInYourBuild]] for help integrating Guava into your build environment.

GWT notes

- **Known issue:** If you see `No source code is available for type java.lang.InterruptedException`, add `<inherits name="java.lang.Lang"/>` to your `.gwt.xml`.
- Incompatible changes to GWT and Guava are coming. Guava 19 works with versions of GWT prior to the forthcoming GWT 2.8.0 but will in many cases *not* work with GWT 2.8.0. Once GWT 2.8.0 is out, we will release Guava 20, which will *require* GWT 2.8.0.
- Additionally, Guava 19 will be the last version to work with the deprecated GWT "classic" Dev Mode. Newer versions will continue to work with Super Dev Mode.
- Guava 19 adds support for many `util.concurrent` APIs under GWT.

J2ObjC notes

- Guava is now annotated for use with the J2ObjC transpiler.

Issues resolved

42 issues are resolved in this release.

API Changes

Full JDiff Report of changes since release 18.0.

Note: for some reason, this diff is showing interface methods as having changed from non-abstract to abstract, resulting in many APIs with no changes showing up as having changes.

Significant API additions and changes

common.base

- Added `CharMatcher` static factory methods equivalent to the `CharMatcher` constants. For example, added `CharMatcher.whitespace()` which is equivalent to `CharMatcher.WHITESPACE`. Eventually, the constants will be deprecated and removed.
  - This is being done because using constants requires a large number of classes to be initialized when anything from `CharMatcher` is used; switching to static factory methods allows classes to be initialized only as needed for the type of `CharMatcher` actually being used.
- Added `Throwables.lazyStackTrace(Throwable)` - Returns a `List<StackTraceElement>` that may load the stack trace elements lazily. Useful if you want to get only the first N elements of the stack trace efficiently.
- Added `lazyStackTraceIsLazy()`- Returns whether or not the above method is able to use the special implementation that makes it lazy on the current platform.
- Added `VerifyException` constructor overloads taking a `Throwable` cause.

common.cache This package has graduated from `@Beta`, making it safe to use in library code.

- Added visibility of `CacheLoader.UnsupportedLoadingOperationException`
- Added `RemovalNotification.create`
  - These should only be needed if creating a custom cache implementation

common.collect Added factory and builder methods for various `ImmutableMap`s and `ImmutableMultimap`s that take `Iterable<Map.Entry>`.

- Added `FluentIterable.toMultiset()`
- Added `RangeSet.asDescendingSetOfRanges()` and `RangeMap.asDescendingMapOfRanges()`
- Added `Lists.cartesianProduct(List...)` and `Lists.cartesianProduct(List<List>>)`
- Added `Maps.newLinkedHashMapWithExpectedSize(int)`
- Re-added `Multisets.removeOccurrences(Multiset, Multiset)` which was (binary incompatibly) missing in 18.0 because it was replaced with `Multisets.removeOccurences(Multiset, Iterable)`
- **Deprecated** `MapConstraint` and `MapConstraints`
- **Deprecated** `Sets.newSetFromMap(Map)` - Java 6 provides `Collections.newSetFromMap(Map)`

- **Removed** `MapMaker.softValues()`

common.eventbus

- Added `EventBus.identifier()`
- **Removed** protected method `AsyncEventBus.dispatchQueuedEvents()` (made package-private)

common.hash

- Added `BloomFilter.create` overloads taking a `long` for the `expectedInsertions`
- Added `Hashing.sha384()`
- Added `Hashing.concatenating(HashFunction, HashFunction, HashFunction...)` and `Hashing.concatenating(Iterable<HashFunction>)`

common.io

- Added `ByteSource.sizeIfKnown()`
- Added `CharSource.length()`
- Added `CharSource.lengthIfKnown()`

common.net

- Added a couple new constants to `HttpHeaders` and `MediaType`
- Updated public suffix list for `InternetDomainName`

common.reflect

- Added `TypeToken.isSubtypeOf(TypeToken)`, `TypeToken.isSupertypeOf(TypeToken)` and overloads of both that take a `Type`
- **Deprecated** `TypeToken.isAssignableFrom(TypeToken)` and `TypeToken.isAssignableFrom(Type)` - `isSupertypeOf` provides equivalent behavior with a less confusing name

common.util.concurrent

- A `CancellationException` from our `ListenableFuture` implementations no longer chains in a stack trace that shows where `cancel` was called. To reenable these stack traces, set system property `guava.concurrent.generate_cancellation_cause` to `true`.
- `Futures.getChecked`, the replacement for `Futures.get` (see below), has become stricter. It now rejects calls that pass an unsuitable exception type, even if the input `Future` succeeded. Previously, it would reject only calls in which the input `Future` had failed.
- Added `AbstractFuture.newCancellationCause()`
- Added `AbstractFuture.setFuture(ListenableFuture)`
- Added `Futures.getChecked`
- Added `Futures.catching` and `Futures.catchingAsync`

- Added `Futures.transformAsync`
- Added `Futures.withTimeout`
- **Deprecated** `FutureFallback` and `Futures.withFallback` methods - these are replaced with `Futures.catching`
- **Deprecated** `Futures.get` methods taking a `Class<X extends Exception>` - these are replaced with `Futures.getChecked`
- **Deprecated** `Futures.transform` methods taking an `AsyncFunction` - these are replaced with `Futures.transformAsync`