# Solución Inicial

# Revisión de Clases de la API de Java

Clases correspondientes a la API de Java que guardan relación con las fechas y horas. No todas las clases serán utilizadas, solo aquellas que hagan función de entidad y que almacenen o puedan devolver información con respecto a las fechas y las horas.

## Que datos se necesitan

* Nombre y paquete de la clase.
* Especificación de la clase (utilidad, comprensión)
* Datos que almacena o puede devolver, ya sea de manera directa o indirecta.
* Jerarquía de la clase (padres e hijos).
* Tipos de datos a los cuales puede ser convertido de manera directa de los cuales se enumere.

## Clases Adicionales que puede que se utilicen utilizar:

Java.time.ZoneId

Java.time.ZoneDateTime

Java.time.format.DateTimeFormatter

java.time.Instant

java.time.temporal.\*

## Nombre completo de clases de la API de Java a utilizar

**Nombre Completo:** Java.util.Calendar

**Especificación:** Clase abstracta que sirve como base para establecer el contexto de un calendario (Calendario Gregoriano (GregorianCalendar) y Calendario Juliano (JulianCalendar)). Almacena la información como un valor long (entero de mayor rango) que permite almacenar los milisegundos que existen entre la primera fecha de la computadora y la fecha establecida. La primera fecha de la computadora es 1 de enero de 1970, esta fecha ha sido almacenada como una constante de tipo long a la cual se le adicionan los milisegundos transcurridos entre esa fecha y la fecha que se desea almacenar. De esta manera se mantiene una aproximación mayor de la fecha real, con la cual es posible realizar cálculos y obtener los campos necesarios. Como java.util.Calendar es una clase abstracta todas las instancias de Calendar son construidas a través de la clase interna Calendar.Builder, la cual construye el calendario siempre como calendario gregoriano (java.util.GregorianCalendar). Calendar es una clase que puede definirse como una línea de tiempo en el espacio temporal.

**Datos que almacena o que puede devolver:** año, mes, día del mes, día del año, día de la semana, semana, hora, AM/PM, milisegundos transcurridos. Para los minutos y segundos no los almacena directamente lo que hace que siempre devuelva 00, para que sea posible la devolución de minutos y segundos la clase tiene que ser de tipo java.util.GregorianCalendar, por defecto siempre lo es.

**Jerarquía:** Padres: ninguno con relación a fechas horas. Hijos: GregorianCalendar.

**Tipos de datos a los cuales puede ser convertido de manera directa:** java.util.Date (método getTime()), java.time.Instant (método toInstant()).

**Nombre Completo:** Java.util.GregorianCalendar.

**Especificación:** SubClase de java.util.Calendar, es el tipo de calendario más utilizado, especifica, los campos utilizados actualmente en el cálculo de fechas. Por defecto los objetos de tipo Calendr creados a través de sus métodos estáticos getInstance(), siempre generan un objeto de tipo GregorianCalendar.

**Datos que almacena o que puede devolver:** Puede devolver los mismos datos que Calendar, además de los minutos y los segundos.

**Jerarquía:** Padres: Calendar Hijos: ninguno.

**Tipos de datos a los cuales puede ser convertido de manera directa:**

Los mismos que java.util.Calendar. y java.util.Calendar a través de herencia.

**Nombre Completo:** Java.util.Date:

**Especificación:** Clase que sirve para almacenar la fecha y hora con una precisión en milisegundos. La clase ha quedado obsoleta por una imprecisión entre el sistema UT y el sistema UTC. Se utiliza la clase DateFormat para la conversión a cadena de caracteres y la clase Calendar es la que debe ser usada para la conversión de fechas y horas. La clase no puede ser utilizada para la obtención directa de sus valores ni debe ser usada para ser instanciada. Solo existe por motivos de compatibilidad y por motivos de utilidad para la persistencia, por ejemplo java.sql.Date es una clase que hereda de java.util.Date. La clase se basa internamente en un tipo de objeto java.util.Calendar, pero no establece las precisiones los diferentes sistemas horarios.

**Datos que almacena o que puede devolver:**

Año, mes día, horas, minutos, segundos, milisegundos.

**Jerarquía:** No hay padres ni hijos que definan en la conversión.

**Tipos de datos a los cuales puede ser convertido de manera directa:**

java.time.Instant (toInstance())

**Nombre Completo:** Java.time.LocalDate

**Especificación:** La clase permite almacenar la fecha sin una zona de tiempo. La clase representa el contexto de año, mes y día sin entender el concepto de horas, minutos y segundos y de línea de tiempo. Se puede entender gráficamente como el calendario de una pared.

**Datos que almacena o que puede devolver:**

Año, mes y día del mes.

**Jerarquía:** Padres:java.time.Temporal, java.time.TemporalAdjuster, java.time.chrono.ChronoLocalDate. Hijos: no tiene.

**Tipos de datos a los cuales puede ser convertido de manera directa:**

Java.time.Temporal, java.time.LocalDateTime, java.time.ZonedDateTime, java.time.OffsetDateTime, java.time.chrono.ISOChronology, java.time.Period

**Nombre Completo:** Java.time.LocalTime

**Especificación:** La clase permite almacenar la hora sin una zona de tiempo. La clase representa el contexto de horas, minutos y segundos, sin entender el concepto de años, meses, semanas, días y sin entender el concepto de línea de tiempo. Se puede entender gráficamente como un reloj de pared, que solo muestra la hora, los minutos y los segundos.

**Datos que almacena o que puede devolver:**

Hora, minutos, segundos, nanosegundos.

**Jerarquía:** Padres:java.time.Temporal, java.time.TemporalAdjuster, Hijos: no tiene.

**Tipos de datos a los cuales puede ser convertido de manera directa:**

Java.time.Temporal, java.time.TemporalAdjuster, java.time.LocalDateTime, java.time.OffsetTime,

**Nombre Completo:** Java.time.LocalDateTime

**Especificación:** La clase equivale a una combinación de LocalDate y LocalTime juntas, no puede representar una línea de tiempo.

**Datos que almacena o que puede devolver:**

Año, mes, día, horas, minutos, segundos, nanosegundos.

**Jerarquía:** Padres:java.time.Temporal, java.time.TemporalAdjuster, java.time.chrono.ChronoLocalDateTime. Hijos: no tiene.

**Tipos de datos a los cuales puede ser convertido de manera directa:**

Java.time.Temporal, java.time.TemporalAdjuster, java.time.LocalDateTime, java.time.ZonedDateTime, java.time.OffsetDateTime, java.time.chrono.ISOChronology, java.time.Instant.