1000 commits de LibreDTE Lib Core: de 2015 a la versión 26.11

La historia de LibreDTE Lib Core contada a través de sus primeros 1000 commits: cómo nació, cuándo apareció cada funcionalidad importante, por qué se rompió la compatibilidad y qué viene ahora.



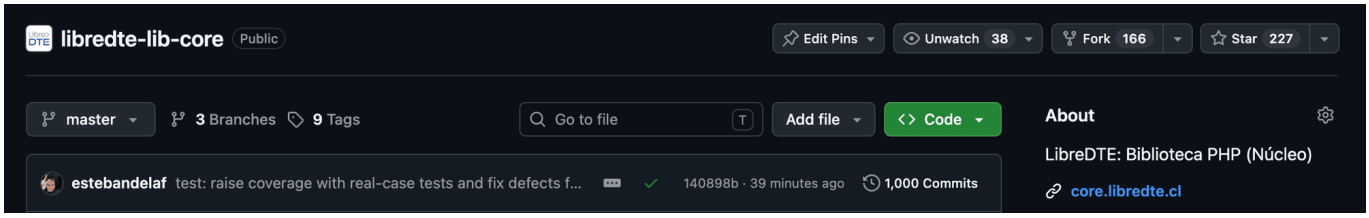
Esteban Delaf · 12/11/2026 · #historia

1000 commits de LibreDTE Lib Core: de 2015 a la versión 26.11

Hoy el repositorio de **LibreDTE Lib Core** llegó al commit número 1000. El primero lo hice el 8 de diciembre de 2014, y lo que subí ese día era mucho más modesto de lo que hoy es el proyecto: una clase para obtener el token de autenticación automática del SII.

Los números redondos son una excusa tan buena como cualquier otra para detenerse a mirar atrás. Así que este artículo cuenta cómo llegamos hasta aquí: qué es Lib Core, qué fue pasando en estos 1000 commits, por qué un día decidí romper la compatibilidad con todo lo anterior y qué viene ahora.

Así se ve el repositorio en GitHub hoy, al llegar al commit 1000: en la rama `master`, con el último commit a la vista, el contador de commits en 1000, y las estrellas y forks que ha ido juntando el proyecto en todos estos años.



La versión **v26.11.0** se marca unos commits después, el 1002, para que quede asociada a este hito.

Es una versión especial por lo simbólico, pero también porque el commit 1000 subió la cobertura de pruebas a más de 95% y dejó el análisis estático en el nivel máximo (nivel 10 de PHPStan). Más abajo explico qué significa y por qué todavía no es una versión “definitiva”.

Qué es LibreDTE Lib Core

LibreDTE tiene por objetivo proveer **facturación electrónica libre, accesible y bien documentada para Chile**. Lib Core es su núcleo: la biblioteca PHP que conversa con el Servicio de Impuestos Internos (SII) para emitir y recibir Documentos Tributarios Electrónicos (DTE).

Con ella se puede construir, validar y firmar documentos tributarios, generar su representación impresa, enviarlos al SII y consultar su estado, intercambiarlos con otras empresas y trabajar con libros, registros tributarios y cesión electrónica. Es software libre bajo licencia AGPL.

Si quieres la versión corta de cómo se usa hoy, está en el artículo de bienvenida y en la documentación.

Un proyecto personal

Antes de la historia hay un dato que la explica casi completa: **LibreDTE siempre fue un proyecto personal**. Lo programé en mis tiempos libres, cuando no estaba haciendo cosas de la empresa. De los 1000 commits, más de 900 los hice yo. El resto vino de personas que aportaron correcciones y funcionalidades a lo largo de los años, y que fueron muy valiosas. Gracias a cada una.

Eso explica algo que se ve a simple vista cuando se mira el gráfico de actividad del repositorio: no es una línea pareja. Son olas.

Año	Commits
2014	5
2015	224
2016	244
2017	83
2018	56
2019	31
2020	40
2021	25
2022	10
2023	11
2024	58

Año	Commits
2025	54
2026 (hasta hoy)	159

Casi la mitad de la historia, 473 commits, ocurrió entre fines de 2014 y 2016. Veamos por qué.

2015 y 2016: construir todo

En esos dos años se construyó lo esencial. Esta es la hoja de ruta de lo que fue apareciendo, tal como quedó en los commits:

Diciembre de 2014 a agosto de 2015: la conexión con el SII. Empezó por lo más básico: pedir el token de autenticación, elegir entre ambiente de certificación y producción, consultar el estado de un envío. Después vinieron la firma electrónica y la generación de XML a partir de arreglos de PHP.

Septiembre de 2015: el primer DTE completo. En pocos días se agregaron las clases para generar un DTE y su sobre de envío, el ejemplo de factura electrónica (tipo 33), la normalización de notas de crédito y débito, la validación de esquemas y firma, y las clases para el proceso de certificación del SII, incluyendo la etapa de intercambio. El 11 de septiembre quedó anotado en el historial como “muestras impresas en PDF fueron aceptadas por el SII”. Esa fue una de las etapas de la certificación.

Certificarse con el SII fue, para ser honesto, un cacho. En ese momento lo hice armando todo con los ejemplos de la biblioteca: en la rama `legacy` todavía existe el directorio `examples/`, con los pasos que pedía la certificación. Armaba cada caso ahí y veía si pasaba. Más adelante, en la aplicación web de LibreDTE (que es otro repositorio), construí una herramienta específica para la certificación.

Octubre a diciembre de 2015: más documentos y libros. Guías de despacho (con su PDF), libros de compras y ventas (IEC/IEV), consumo de folios, compatibilidad con PHP 7 y, el 12 de diciembre, el soporte de **boletas electrónicas** (tipo 39). También aparecieron los primeros comandos de consola, como la generación de PDF.

Enero a diciembre de 2016: pulir y ampliar. El 25 de enero de 2016 el historial anota que “se certificaron boletas electrónicas”. Siguieron la factura de compra (febrero), los documentos de exportación (abril), el soporte para escribir documentos en YAML (junio), mejoras al PDF (papel continuo, logo, períodos de facturación, tablas de libros) y, en diciembre, los esquemas y el código de la **cesión electrónica** para factoring, que llegó gracias a un aporte de la comunidad.

Esos dos años definieron lo que hoy sigue siendo el corazón del proyecto: emitir, firmar, enviar, intercambiar, informar y ceder documentos.

2017 a 2019: consolidar

Con lo esencial funcionando, el ritmo bajó y el trabajo cambió de naturaleza. Fueron años de ajustes finos: PHPUnit con namespace (2017), la posibilidad de crear XML con namespace, nuevos casos de normalización para boletas, referencias, el registro de compras y ventas (RCV) del SII, el medio de pago en el PDF, una clase de PDF que se puede extender para hacer diseños personalizados (2018) y métodos para saber cuánto tiempo le queda a una firma electrónica (2019).

2020 a 2023: un parón que no fue problema

En 2019 contraté personas a fin de año, y con la pandemia de 2020 el negocio creció muchísimo. Desde ese momento tuve que dedicarme al 100% a la empresa, y LibreDTE dejó de ser lo que ocupaba mis tardes y fines de semana.

La buena noticia es que para entonces **la biblioteca ya estaba madura**. Tenía todos los cambios relevantes y no se vio afectada por la baja de ritmo. Lo que vino después fueron necesidades puntuales:

- Julio de 2020: se numeran versiones, empezando por la 20.07, y se pueden obtener las cesiones desde un AEC.
- Octubre de 2020: nuevo esquema de boletas y desglose de IVA (versión 20.10).
- 2021: correcciones al token del RCV, decimales según la moneda en los totales, vencimiento del CAF y vigencia de la firma (versión 21.10).
- 2022 y 2023: soporte de PHP 7.3 y 7.4, nuevas referencias, ajustes del DV en mayúscula, la validación de la firma y traducciones de errores de esquema XML.

Entre 2020 y 2023 hubo 86 commits en total, contra los 468 de 2015 y 2016. Y eso, en este caso, fue una buena señal: significaba que lo importante ya estaba hecho.

2024: el quiebre

En 2024 volví a trabajar en LibreDTE en serio, pero para hacer algo distinto. El 9 de mayo de 2024 hice el commit que inició la **migración a un nuevo namespace y a PHP 8**. Avisé en ese mismo mensaje que la rama no iba a funcionar hasta el final.

¿Por qué romper la compatibilidad? Porque el diseño de la biblioteca inicial era **muy malo**. Lo digo sin eufemismos, y con todo cariño: lo escribí en 2015, con rapidez, para resolver un problema urgente, y funcionó muy bien para eso. Pero con los años las limitaciones eran evidentes: todo estaba dentro de una sola carpeta, acoplado al SII y a Chile, difícil de probar y de extender. Si te interesa el detalle técnico, lo conté en De un monorepo a un ecosistema.

Parchar no alcanzaba. Para arreglarlo de verdad había que cambiar la forma en que está construida, y eso no se puede hacer sin romper cosas. Esto pasó con la identidad de la biblioteca:

	Versión anterior	Versión nueva
Paquete en Composer	`sasco/libredte`	`libredte/libredte-lib-core`
Namespace	`sasco\LibreDTE`	`libredte\lib\Core`
PHP	7.3	8.x (hoy 8.5)

La versión anterior no desapareció. Quedó en la **rama `legacy`** del repositorio y sigue recibiendo los cambios mínimos para que quien la use siga funcionando. Por ejemplo, la versión 24.1.2 se publicó hace poco en esa rama. Si hoy usas LibreDTE y funciona, no tienes que migrar apurado. Cuando quieras hacerlo, lo recomendable es planificarlo como una integración nueva, no como una actualización.

2024 a 2026: la nueva biblioteca

Reescribir el núcleo me tomó **más de dos años**. Si te preguntas por qué tanto: porque sigue siendo un trabajo de mis tiempos libres, fuera de las obligaciones de la empresa. Y porque desde 2024 gran parte de mi tiempo fue a los paquetes `derafu/*`, que son en buena medida la base de la nueva LibreDTE. Por eso el historial de este repositorio tiene huecos, por ejemplo entre marzo y julio de 2025: no estuvo parado, estaba trabajando en los paquetes de Derafu de los que Lib Core depende. Esta es la hoja de ruta de lo que fue apareciendo:

Septiembre de 2024: la base nueva. El 23 de septiembre se subió la primera versión de la biblioteca con el nuevo namespace, refactorizada con prácticas actuales de PHP 8, con integración continua, análisis estático y documentación. Se agregaron también casos de uso por tipo de documento, útiles para aprender y para probar.

Diciembre de 2024: renderizado y entrada de datos. Llegó la base para generar los documentos en HTML y PDF con plantillas, los *parsers* que convierten distintos formatos de entrada en un documento y un validador de datos.

Enero de 2025: reestructuración sobre Derafu. Gran parte del código genérico (arreglos, fechas, XML, certificados, caché, configuración) se movió a paquetes independientes `derafu/*`. Con eso nacieron también el procesamiento en lote (por ejemplo, emisión masiva desde un CSV), el soporte para monedas de Aduana y el componente de intercambio de documentos. El camino lo cuento en De un monorepo a un ecosistema.

Agosto y septiembre de 2025: Backbone. La biblioteca se reorganizó sobre `derafu/backbone`, lo que permite agregar tipos de documentos, formatos o canales sin tocar el núcleo, y se definieron los recursos de la API. Por qué eso importa está en Extender sin tocar el núcleo.

Marzo a junio de 2026: lo que faltaba de lo que la versión anterior ya hacía. El libro de boletas, las acciones con el RCV, las respuestas del proceso de intercambio, la cesión electrónica, la gestión de los CAF, la generación de libros de compra y venta sin detalle, y el paso a PHP 8.5. También se cambió la biblioteca para generar los códigos PDF417 de los timbres.

Agosto a octubre de 2026: ejemplos y operaciones. Se agregaron ejemplos de operaciones, nuevos campos para guías de despacho, la actualización del esquema XML de DTE, el código en inglés y un trabajo grande en pruebas de extremo a extremo con paquetes externos que prueban esta biblioteca en conjunto.

Noviembre de 2026: calidad. En pocas semanas el análisis estático pasó de un nivel bajo al **nivel 10 de PHPStan**, el máximo, y en el commit 1000 la cobertura de pruebas superó el **95%**. Esas pruebas con casos reales, además, encontraron varios defectos que se corrigieron.

Lo que viene ahora

La versión v26.11.0 todavía depende de paquetes en desarrollo (`dev-main`). La marco igual por coincidir con el hito de los 1000 commits, pero no es una versión estable en el sentido de que cualquiera pueda instalarla con tranquilidad. Lo que sigue apunta a dos cosas.

a) Una versión estable que no dependa de `dev`

Hoy Lib Core depende de varios paquetes `derafu/*` en `dev-main`, y todos son míos. Para publicar una versión realmente estable:

1. Los paquetes `derafu/*` de los que depende deben tener versiones publicadas, con su propia API revisada y sus pruebas al día.
2. Lib Core debe pasar a depender de esas versiones y no de `dev-main`.
3. Debo probar que todo está estable de punta a punta: emisión, envío, intercambio, libros, cesión y las demás operaciones, con ambientes de certificación y producción del SII.

No pondré fecha. Prefiero publicarla cuando esté bien probada.

b) Más cobertura y mejor calidad de pruebas

Que la cobertura sea alta no significa que las pruebas sean buenas: una línea puede ejecutarse sin que ninguna prueba compruebe lo que hace. Para mirar la calidad y no solo la cantidad, voy a usar **pruebas de mutación** (con Infection). La técnica consiste en alterar el código a propósito, por ejemplo cambiando un `>` por un `>=`, y ver si alguna prueba falla. Si ninguna falla, ese comportamiento no estaba realmente probado.

La idea es subir la cobertura todavía más y revisar los mutantes que sobreviven, para agregar las pruebas que falten.

Gracias

Gracias a quienes usan LibreDTE, a quienes reportan problemas, a quienes aportaron código y a quienes han esperado esta nueva versión con paciencia. Todavía queda trabajo, pero el camino de estos 1000 commits muestra algo que me gusta: lo que se construye con constancia, aunque sea en

ratos libres, termina llegando.

El código está en GitHub, el paquete en Packagist y la documentación en core.libredte.cl/docs.
Síguenos para no perderte lo que viene.

Última actualización el 12/11/2026