



License MIT

node `>=18.0.0`

Una potente colección de scripts para generar y gestionar tokens de diseño para tu sistema de diseño. Cada maguito se especializa en crear tipos específicos de tokens, garantizando consistencia y eficiencia en tus proyectos.



Primeros Pasos

1. Instalar Node.js

Descarga e instala [Node.js](#) en tu ordenador.

2. Instalar VS Code

Descarga e instala [Visual Studio Code](#) para una experiencia de desarrollo mejorada.

3. Abrir Terminal

- **VS Code:** Presiona `Ctrl + `` (Windows/Linux) o `Cmd + `` (Mac)
- **Terminal del Sistema:**
 - Windows: `Windows + R`, escribe `cmd`
 - Mac: `Command + Space`, escribe `terminal`
 - Linux: `Ctrl + Alt + T`

4. Descargar/Clonar el Repositorio

[Descargar ZIP](#)

o

Clonar Repo

```
git clone https://github.com/fulviabuonanno/design-tokens-wizards.git
cd design-tokens-wizards
```


5. Instalar Dependencias

```
npm install
```

6. Ejecutar los Scripts

Elige entre los siguientes asistentes:

Maguito de Tokens	Nombre Script	Comando	Descripción	Versión
 COLOR WIZ	<code>color-wiz.js</code>	<code>npm run color</code>	Generar y gestionar tokens de color	2.6.0 
 TYPOGRAPHY WIZ	<code>typo_wiz.js</code>	<code>npm run typo</code>	Generar y gestionar tokens tipográficos	1.2.1 
 SPACE WIZ	<code>space_wiz.js</code>	<code>npm run space</code>	Generar y gestionar tokens de espaciado	1.7.0 
 SIZE WIZ	<code>size_wiz.js</code>	<code>npm run size</code>	Generar y gestionar tokens de tamaño	1.7.0 
 BORDER RADIUS WIZ	<code>radii_wiz.js</code>	<code>npm run radii</code>	Generar y gestionar tokens de radio	1.7.0 

Hechizo	Nombre Script	Comando	Descripción	Versión
MERGE SPELL	<code>merge_spell.js</code>	<code>npm run merge</code>	Combinar todos los archivos de tokens en uno	1.2.0
CLEAR SPELL	<code>clear_spell.js</code>	<code>npm run clear</code>	Eliminar todos los archivos generados de una vez	1.2.1 

Leyenda:

 Parche //  Cambio Menor //  Cambio Mayor



Estructura del Proyecto

```
src/  
  wizards/      # Todos los scripts de asistentes (color, typo, space, size, rad:  
  spells/       # Scripts de utilidad (merge, clear)  
  config/       # Configuración y scripts auxiliares  
output_files/   # Donde se guardan los tokens generados  
  tokens/  
    color/  
    typography/  
    space/  
    size/  
    border-radius/  
css/  
scss/  
final/
```

Maguito de Tokens de Color ✨

Versión 2.6.0

Gestionar tokens de color puede parecer tan mágico como dominar la alquimia, pero con el Maguito de Tokens de Color 🧙‍♂️, tu viaje para crear una paleta luminosa es muy sencillo. Comienza con un tono base que establecerá el tono mágico, y el maguito te guiará a través de la creación de un espectro deslumbrante de tokens. Ya sea que estés definiendo un tono distintivo de marca o curando un sistema de color completo, deja que este maguito transforme tu visión creativa en una realidad vibrante.

1. Invoca el Hechizo

Comienza tu viaje ejecutando el script del Maguito de Tokens de Color. Abre tu terminal y lanza el hechizo con:

```
npm run color
```

2. Selecciona el Tipo de Token

Elige entre colores globales y semánticos:

- **Global Colors:** Colores base que forman la base de tu sistema de diseño
- **Semantic Colors:** (Próximamente) Colores basados en su significado y propósito

Nota: El maguito actualmente soporta **colores globales**. Los colores semánticos están en desarrollo y estarán disponibles en una actualización futura.

3. Define la Categoría Global (Opcional)

Organiza tus colores en grupos lógicos:

- primitives
- foundation

- core
- basics
- essentials
- global
- roots
- custom

4. Establece el Nivel de Nomenclatura (Opcional)

Proporciona contexto sobre cómo se debe usar el color:

- color
- colour
- palette
- scheme
- custom

5. Nombra tu Color

Ingresa un nombre para tu color (ej., `blue` , `yellow` , `red`).

6. Selecciona el Color Base

Proporciona un código de color HEX (ej., `#FABADA`) para usar como tu color base.

7. Elige el Tipo de Escala

Selecciona cómo quieres generar tus paradas de color:

- **Incremental:** Genera paradas usando un paso incremental
 - 10 en 10 (ej., 10, 20, 30, 40)
 - 25 en 25 (ej., 25, 50, 75, 100)
 - 50 en 50 (ej., 50, 100, 150, 200)
 - 100 en 100 (ej., 100, 200, 300, 400)
- **Ordinal:** Crea una secuencia
 - Con relleno (ej., 01, 02, 03, 04)
 - Sin relleno (ej., 1, 2, 3, 4)
- **Alfabético:** Usa letras

- Mayúsculas (ej., A, B, C, D)
- Minúsculas (ej., a, b, c, d)
- **Stops Semánticas:** Genera paradas semánticas
 - 1 stop: base
 - 2 stops: dark, base, light
 - 4 stops: darker, dark, base, light, lighter
 - 6 stops: darkest, darker, dark, base, light, lighter, lightest
 - 8 stops: ultra-dark, darkest, darker, dark, base, light, lighter, lightest, ultra-light
 - 10 stops: ultra-dark, darkest, darker, dark, semi-dark, base, semi-light, light, lighter, lightest, ultra-light

8. Establece el Número de Stops

Especifica cuántos stops de color quieres generar (1-20).

9. Personaliza los Rangos de Color

Define cómo tus colores se mezclan con blanco y negro:

- **Rango Predeterminado:** 10% a 90% de mezcla
- **Rango Personalizado:** Define tus propios porcentajes de mezcla (0-100%)
 - Valores más bajos crean variaciones más extremas de claro/oscuro
 - Valores más altos crean variaciones más sutiles
 - Valores de 0% o 100% resultarán en colores puramente blancos/negros

10. Vista Previa de los Tokens

El maguito mostrará la vista previa de tu color junto con la tabla de paradas. El nombre del token se construirá basado en tus selecciones:

Tipo: Global

Categoría: primitives (opcional)

Nivel: color (opcional)

Nombre: blue

Escala	HEX	Muestra

base	#FABADA	[✨]
01	#F0E3D2	[✨]
02	#E6D6BF	[✨]
...

11. Genera tus Artefactos

Una vez confirmado, el maguito:

- Exportará tus tokens en formato JSON de Tokens Studio Almacenado en: `output_files/tokens/color` como `color_tokens_{format}.json`
- Creará archivos CSS y SCSS con tus tokens como variables Almacenado en `output_files/tokens/css/` y `output_files/tokens/scss/` respectivamente como: `color_variables_{format}.css` y `color_variables_{format}.scss`

12. Expande tu Paleta

Añade más colores a tu paleta repitiendo el proceso.

13. Finaliza el Hechizo

Revisa los archivos de salida e integra tus tokens de diseño en tu sistema.

Nota: El maguito actualmente soporta solo colores globales. Los colores semánticos, categorías y niveles de nomenclatura están en desarrollo y estarán disponibles en una actualización futura.

¡Deja que el arte de la creación de tokens infunda tu proyecto con creatividad sin fin—y que tus colores brillen para siempre!

Maguito de Tokens de Typography ✨

Versión 1.2.0

El Maguito de Tokens de Typography 🧙 es tu compañero encantado para crear un sistema tipográfico completo. Como un hábil alquimista mezclando ingredientes potentes, este maguito te ayuda a combinar familias de fuentes, tamaños, pesos, espaciados y alturas en un elixir tipográfico armonioso para tu sistema de diseño.

1. Invoca el Hechizo

Comienza tu viaje tipográfico ejecutando el script del Maguito de Tokens Tipográficos:

```
npm run typo
```

2. Elige tus Propiedades

Selecciona qué propiedades tipográficas deseas configurar:

- Font Families
- Font Sizes
- Font Weights
- Letter Spacing
- Line Height

3. Para Cada Propiedad:

Font Family

- Nombra tu propiedad (fontFamily, font-family, fonts, ff, o personalizado)
- Define 1-3 font families con alternativas
- Elige convención de nombres:
 - Semantic (primary, secondary, tertiary)

- Purpose-based (heading, body, detail)
- Ordinal (1, 2, 3)
- Alphabetic (a, b, c)

Font Size

- Nombra tu propiedad (fontSize, font-size, size, fs, o personalizado)
- Selecciona tipo de escala:
 - 4-Point Grid
 - 8-Point Grid
 - Modular Scale
 - Custom Intervals
 - Fibonacci Scale
- Elige unidad (px, rem, em)
- Define 1-12 tamaños con convención de nombres:
 - Size (xs, sm, md, lg, xl)
 - Incremental (10, 20, 30)
 - Ordinal (1, 2, 3)
 - Alphabetic (a, b, c...)

Font Weight

- Nombra tu propiedad (fontWeight, font-weight, weight, fw, o personalizado)
- Selecciona de pesos estándar (100-900)
- Elige convención de nombres:
 - Size (xs to xl)
 - Semantic (thin to bold)
 - Ordinal (1 to 5)
 - Purpose-based (body, heading...)

Letter Spacing

- Nombra tu propiedad (letterSpacing, letter-spacing, tracking, ls, o personalizado)
- Elige tipo de escala:
 - **Default Scale:** (-1.25 to 6.25)
 - **Custom Values:** Define tus propios valores
- Selecciona unidad (em, rem, %)
- Define 1-7 valores con convención de nombres:
 - Size (xs to xl)
 - Incremental (100, 200...)
 - Ordinal (01, 02... or 1, 2...)
 - Alphabetic (a, b, c...)

Line Height

- Nombra tu propiedad (lineHeight, line-height, leading, lh, o personalizado)
- Elige tipo de escala:
 - **Default Scale 1:** (1.1, 1.25, 1.5, 1.6, 1.75, 2.0)
 - **Default Scale 2:** (1.0, 1.2, 1.5, 1.6, 2.0)
 - **Custom Values:** Define tus propios valores
- Elige convención de nombres:
 - Size (xs to xl)
 - Semantic (tight, normal, loose, relaxed, spacious)
 - Ordinal (1 to 5)
 - Purpose-based (body, heading, display, compact, expanded)
 - Incremental (100, 200...)
 - Alphabetic (a, b, c...)

4. Vista Previa de los Tokens

Para cada propiedad, verás una tabla de vista previa mostrando tus valores configurados:

Property: Font Size

Scale	Value
xs	12px
sm	14px
md	16px
lg	18px
xl	20px

5. Genera tus Artefactos

Una vez confirmado, el maguito:

- Exportará tus tokens en formato JSON de Tokens Studio Almacenado en:
`output_files/tokens/typography/typography_tokens.json`
- Creará archivos CSS y SCSS con tus tokens como variables Almacenado en
`output_files/tokens/css/typography/typography_variables.css` y
`output_files/tokens/scss/typography/typography_variables.scss`

6. Finaliza el Hechizo

Revisa los archivos de salida e integra tus tokens tipográficos en tu sistema.

⚠ IMPORTANTE: Cada paso incluye directrices de accesibilidad y recomendaciones para asegurar que tu sistema tipográfico sea tanto hermoso como funcional. El maguito actúa como tu consejero de confianza, sugiriendo valores óptimos mientras te da la libertad de personalizar según tus necesidades.

■ Maguito de Tokens de Spacing ✨

Versión 1.7.0

Crear y gestionar tokens de espaciado puede ser una tarea desalentadora, pero con el Maguito de Tokens de Spacing 🧙 , puedes agilizar el proceso y ahorrar un tiempo precioso. Comienza con un valor base de espaciado, y el maguito generará una variedad de tokens de espaciado en diferentes unidades, listos para integrarse en tu sistema de diseño y proyectos de desarrollo.

1. Invoca el Hechizo

Comienza tu viaje ejecutando el script del Maguito de Tokens de Espaciado:

```
npm run space
```

2. Define la Unidad Base

La unidad base predeterminada para los tokens de espaciado es pixels (px).

3. Nombra tus Tokens de Espaciado

Proporciona un nombre para tus tokens de espaciado (ej., space, spc).

4. Selecciona una Escala

Elige una escala predefinida para tus tokens:

- 4-Point Grid System
- 8-Point Grid System
- Modular Scale (based on multiplier)
- Custom Intervals
- Fibonacci Scale

5. Define el Número de Valores

Especifica cuántos valores de espaciado quieres generar (ej., 6 valores para una

escala de pequeño a grande).

6. Elige Criterios de Nomenclatura

Selecciona un patrón de nombres para tus tokens de espaciado:

- Sizes (xs, sm, md, lg, xl)
- Incremental Numbers (100, 200, 300)
- Ordinal Numbers (1, 2, 3)
- Alphabetic (A, B, C or a, b, c)

7. Vista Previa de los Tokens

El maguito mostrará la vista previa de tus tokens de espaciado:

Name: Space

Scale	Value
01	16px
02	24px
03	32px
04	40px

8. Genera tus Artefactos

Una vez confirmado, el maguito:

- Exportará tus tokens en formato JSON de Tokens Studio Almacenado en:
`output_files/tokens/space/space_tokens_{unit}.json`
- Creará archivos CSS y SCSS con tus tokens como variables Almacenado en
`output_files/tokens/css/space/space_variables_{unit}.css` y
`output_files/tokens/scss/space/space_variables_{unit}.scss`

9. Finaliza el Hechizo

Revisa los archivos de salida e integra tus tokens de espaciado en tu sistema.

Maguito de Tokens de Size ✨

Versión 1.7.0

Gestionar tokens de Size puede ser desalentador, pero con el Maguito de Tokens de Tamaño 🧙 , puedes agilizar el proceso y ahorrar un tiempo valioso. Comienza con un tamaño base, y deja que el maguito genere una gama de tokens de tamaño en varias unidades, listos para integrarse en tu sistema de diseño y proyectos de desarrollo.

1. Invoca el Hechizo

Comienza tu viaje ejecutando el script del Maguito de Tokens de Tamaño:

```
npm run size
```

2. Define la Unidad Base

La unidad base predeterminada para los tokens de tamaño es pixels (px).

3. Nombra tus Tokens de Tamaño

Proporciona un nombre para tus tokens de tamaño (ej., size, dimension).

4. Selecciona una Escala

Elige entre las escalas predefinidas:

- 4-Point Grid System
- 8-Point Grid System
- Modular Scale (based on multiplier)
- Custom Intervals
- Fibonacci Scale

5. Define el Número de Valores

Especifica cuántos valores de tamaño quieres generar (ej., 6 valores para una escala de pequeño a grande).

6. Elige Criterios de Nomenclatura

Selecciona un patrón de nombres para tus tokens de tamaño:

- Sizes (xs, sm, md, lg, xl)
- Incremental Numbers (100, 200, 300)
- Ordinal Numbers (1, 2, 3)
- Alphabetic (A, B, C or a, b, c)

7. Vista Previa de los Tokens

El maguito mostrará la vista previa de tus tokens de tamaño:

Name: Size

Scale	Value
01	16px
02	24px
03	32px
04	40px

8. Genera tus Artefactos

Una vez confirmado, el maguito:

- Exportará tus tokens en formato JSON de Tokens Studio Almacenado en:
`output_files/tokens/size/size_tokens_{unit}.json`
- Creará archivos CSS y SCSS con tus tokens como variables Almacenado en
`output_files/tokens/css/size/size_variables_{unit}.css` y
`output_files/tokens/scss/size/size_variables_{unit}.scss`

9. Finaliza el Hechizo

Revisa los archivos de salida e integra tus tokens de tamaño en tu sistema.

Maguito de Tokens de Border Radius ✨

Versión 1.7.0

Crear tokens de border radius se simplifica con el Maguito de Tokens de Border Radius 🧙. Este maguito te guía a través de la definición y generación de tokens de radio de borde para tu sistema de diseño, listos para ser utilizados en varios formatos incluyendo JSON, CSS y SCSS.

1. Invoca el Hechizo

Comienza tu viaje ejecutando el script del Maguito de Tokens de Radio de Borde:

```
npm run radii
```

2. Define el Nombre del Token

Elige un nombre para tus tokens de radio de borde (ej., `border-radius`, `corner-radius`, `radius`).

3. Selecciona Estructura de Escala

Decide si incluir valores 'none' y 'full' para el radio de borde.

4. Define Pasos Intermedios

Opcionalmente, añade pasos intermedios para una escala más granular:

- Sizes (xs, sm, md, lg, xl)
- Incremental Numbers (100, 200, 300, 400)
- Ordinal Numbers (01, 02, 03, 04 or 1, 2, 3, 4)
- Alphabetic (A, B, C, D or a, b, c, d)
- Semantic (subtle, soft, moderate, bold)

5. Define Escala de Valores

Selecciona una escala de valores basada en bases mínimas o expresivas:

- 4-Point Grid System
- 8-Point Grid System
- Modular Scale (based on multiplier)
- Custom Intervals
- Fibonacci Scale

6. Vista Previa de los Tokens

El maguito mostrará la vista previa de tus tokens de radio de borde:

Name: Border Radius

Scale	Value
None	0px
01	4px
02	8px
03	12px
Full	9999px

7. Genera tus Artefactos

Una vez confirmado, el maguito:

- Exportará tus tokens en formato JSON de Tokens Studio Almacenado en:
`output_files/tokens/border-radius/border_radius_tokens_{unit}.json`
- Creará archivos CSS y SCSS con tus tokens como variables Almacenado en
`output_files/tokens/css/border-radius/border_radius_variables_{unit}.css` y
`output_files/tokens/scss/border-radius/border_radius_variables_{unit}.scss`

8. Finaliza el Hechizo

Revisa los archivos de salida e integra tus tokens de radio de borde en tu sistema.

Hechizo de Clear ✨

Versión 1.2.1

El Hechizo de Clear es una herramienta simple pero poderosa para mantener tu directorio de proyecto limpio y organizado. Elimina todos los archivos generados, asegurando que puedas comenzar de nuevo sin ningún desorden.

1. Ejecuta el Script del Hechizo de Limpieza

Abre tu terminal y ejecuta el siguiente comando:

```
npm run clear
```

2. Lo que Sucede Después

- Se eliminan todos los archivos de tokens generados en el directorio `output_files/tokens/`.
- Se eliminan los archivos CSS y SCSS en los directorios `output_files/tokens/css/` y `output_files/tokens/scss/`.
- También se eliminarán los archivos de salida final en el directorio `output_files/final/`, incluyendo archivos JSON, CSS y SCSS.
- Tu directorio de proyecto queda limpio, libre de archivos desactualizados o innecesarios.

Hechizo de Merge ✨

Versión 1.2.0

El Hechizo de Merge está diseñado para simplificar tu flujo de trabajo combinando múltiples archivos de tokens en un único archivo unificado. Esto hace que la gestión e integración de tokens de diseño entre categorías (ej., color, tamaño, espaciado) sea mucho más fácil.

1. Ejecuta el Script del Hechizo de Fusión

Abre tu terminal y ejecuta el siguiente comando:

```
npm run merge
```

2. Lo que Sucede Después

- Todos los archivos de tokens en el directorio `output_files/final/` se combinan en un único archivo JSON, CSS y SCSS.
- El archivo resultante está estructurado de manera consistente, asegurando una integración perfecta en tu sistema de diseño o flujo de trabajo de desarrollo.
- Puedes seleccionar qué formatos y opciones de tus tokens generados quieres incluir en el archivo final combinado.

Creado con Amor en Barcelona por Fulvia Buonanno ✨❤



Descubre más sobre nuestros maguitos en: [Sitio Web de Design Tokens Wizards](#)

Si eres apasionado por los sistemas de diseño y tokens, esta herramienta es tu compañera perfecta, permitiéndote crear tokens sin esfuerzo. Para los fanáticos de los RPG o JRPG, esta herramienta evocará una sensación de nostalgia, mezclando vibraciones clásicas de juegos con tu flujo de trabajo de diseño. 🧩

Creado con amor por Fulvia Buonanno, una Diseñadora de Design Systems basada en Barcelona, esta herramienta tiene como objetivo cerrar la brecha entre el diseño y el desarrollo, haciendo que los tokens sea más accesibles, especialmente para los aventureros recién llegados a este mundillo. 🧙

Dependencias ✨

A continuación se muestra una lista completa de todas las dependencias utilizadas en este proyecto:

Dependencia	Versión	Descripción	Repositorio
chalk	^5.4.1	Estilizado de cadenas de terminal hecho correctamente	chalk/chalk
cli-table3	^0.6.5	Tablas unicode bonitas para la línea de comandos	cli-table3
inquirer	^12.4.2	Una colección de interfaces de usuario comunes para línea de comandos	SBoudrias/Inquirer.js
path	^0.12.7	Módulo path de Node.js	nodejs/node
tinycolor2	^1.6.0	Manipulación y conversión de colores rápida y ligera	bgrins/TinyColor
puppeteer	^20.0.0	API de Chrome sin cabeza para Node.js para automatizar interacciones web	puppeteer/puppeteer



Licencia

Este proyecto está licenciado bajo la Licencia MIT. Esto significa que eres libre de usar, modificar y distribuir el software siempre que se incluya el aviso de copyright original y el aviso de permiso en todas las copias o partes sustanciales del software.

Para más detalles, puedes leer el texto completo de la licencia en el archivo [LICENSE](#) incluido en este repositorio o visitar la Open Source Initiative para más información.

? Solución de Problemas y FAQ

P: ¿Cómo puedo proporcionar comentarios o reportar problemas?

R: ¡Bienvenidos sean tus comentarios! Puedes:

- Contactarme en el [sitio web](#) de los maguitos
- Completar este [formulario](#) para brindarme tu feedback

Tus comentarios me ayudan a mejorar la herramienta y hacerla mejor para todos. Estoy particularmente interesada en:

- Reportes de errores
- Solicitudes de características
- Mejoras en la documentación
- Comentarios sobre la experiencia de usuario
- Problemas de rendimiento

P: ¿Qué versión de Node.js necesito?

R: Necesitas Node.js versión 18.0.0 o superior. Puedes verificar tu versión actual ejecutando:

```
node --version
```

P: ¿Cómo puedo actualizar a la última versión?

R: Para actualizar a la última versión, simplemente ejecuta:

```
npm install design-tokens-wizards@latest
```

P: ¿Puedo usar los tokens generados con cualquier framework?

R: ¡Sí! Los tokens se generan en formatos estándar (JSON, CSS, SCSS) que son

compatibles con la mayoría de los frameworks y herramientas de desarrollo. Los archivos generados pueden ser utilizados con:

- React
- Vue
- Angular
- Svelte
- Y cualquier otro framework que soporte CSS/SCSS

P: ¿Cómo puedo contribuir al proyecto?

R: ¡Me encanta que quieras contribuir! Puedes:

1. Hacer fork del repositorio
2. Crear una rama para tu feature (`git checkout -b feature/AmazingFeature`)
3. Commit tus cambios (`git commit -m 'Add some AmazingFeature'`)
4. Push a la rama (`git push origin feature/AmazingFeature`)
5. Abrir un Pull Request

Para más detalles, consulta [CONTRIBUTING.md](#).

P: ¿Qué hago si encuentro un bug?

R: Si encuentras un bug, por favor:

1. Verifica que no sea un problema de configuración
2. Revisa los issues existentes para ver si ya ha sido reportado
3. Si es un nuevo bug, crea un nuevo issue con:
 - Descripción clara del problema
 - Pasos para reproducirlo
 - Comportamiento esperado vs actual
 - Capturas de pantalla si es relevante
 - Información de tu entorno (OS, Node.js version, etc.)

P: ¿Los tokens generados son compatibles con Tokens Studio?

R: ¡Sí! Los tokens se generan en formato JSON compatible con Tokens Studio, lo que significa que puedes:

- Importar los tokens directamente en Tokens Studio
- Exportar desde Tokens Studio y usar los maguitos para procesarlos
- Mantener la consistencia entre tu sistema de diseño y código

P: ¿Puedo personalizar los formatos de salida?

R: Actualmente, los maguitos generan archivos en:

- JSON (formato Tokens Studio)
- CSS (variables CSS)
- SCSS (variables SCSS)

Si necesitas un formato específico, puedes:

1. Usar el formato JSON y convertirlo a tu formato deseado
2. Solicitar el nuevo formato a través de un issue
3. Contribuir con un nuevo formato

Contacto y Soporte

Para preguntas, sugerencias o reportes de errores, por favor abre un issue o [contáctame](#).

Contribuir

¡Las contribuciones, issues y solicitudes de características son bienvenidas!
No dudes en revisar la [página de issues](#) o enviar un pull request.

Por favor, consulta [CONTRIBUTING.md](#) para las directrices.
