Contrato de Subasta Segura en Ethereum

Descripción

Este contrato inteligente implementa una subasta segura y transparente en la red Ethereum. Permite a los usuarios ofertar por un artículo, gestionar depósitos, extender el tiempo de subasta automáticamente y asegura el reembolso de depósitos a los participantes no ganadores, descontando una comisión del 2%. El propietario puede finalizar la subasta, cancelar antes de recibir ofertas y retirar los fondos de la oferta ganadora.

Variables Principales

- **owner (address, public): Dirección del propietario de la subasta.**
- **auctionEndTime (uint, public): Momento de finalización de la subasta.**
- **maxExtensionTime (uint, public): Tiempo máximo de extensión permitido (por defecto 7 días).**
- **extendedTime (uint, public): Tiempo total extendido.**
- **highestBidder (address, public): Dirección del mejor postor actual.**
- **highestBid (uint, public): Valor de la mejor oferta actual.**
- **bidHistory (Bid[]): Historial de todas las ofertas (accesible mediante paginación).**
- **deposits (mapping, public): Depósitos de cada usuario.**
- **lastBid (mapping, public): Última oferta de cada usuario.**
- **ended (bool, public): Indica si la subasta ha finalizado.**
- **cancelled (bool, public): Indica si la subasta fue cancelada.**

Estructura

Bid

- **bidder (address): Dirección del ofertante.**
- **amount (uint): Monto ofertado.**

Eventos

- **NewBid(address bidder, uint amount): Emitido cuando se realiza una nueva oferta.**
- **AuctionEnded(address winner, uint winningAmount): Emitido cuando finaliza la subasta.**
- **PartialWithdrawal(address bidder, uint amount): Emitido cuando un usuario retira el exceso de su depósito.**
- **DepositWithdrawn(address bidder, uint amount, uint fee): Emitido cuando un usuario retira su depósito (menos comisión).**
- **AuctionCancelled(): Emitido si la subasta es cancelada antes de recibir ofertas.**
- **FeeTransferred(address to, uint amount): Emitido cuando se transfiere la comisión al propietario.**
- **DepositWithdrawnOnCancel(address bidder, uint amount): Emitido cuando un usuario retira su depósito tras la cancelación.**

Funcionalidades Principales

- **Constructor: Inicializa la subasta con la duración en minutos. El propietario es quien despliega el contrato.**
- **bid(): Permite a los usuarios ofertar. La oferta debe ser al menos 5% mayor que la actual. Si la oferta es dentro de los últimos 10 minutos, extiende la subasta (hasta un máximo de 7 días).**
- **partialWithdraw(): Permite a los usuarios retirar el exceso de depósito sobre su última oferta válida durante la subasta.**
- **withdrawDeposit(): Permite a los no ganadores retirar su depósito menos una comisión del 2% después de la subasta.**
- **withdrawDepositOnCancel(): Permite a los usuarios retirar su depósito si la subasta fue cancelada.**
- **endAuction(): Permite al propietario finalizar la subasta manualmente antes del tiempo límite.**

- **withdrawFunds(): Permite al propietario retirar la oferta ganadora después de la subasta.**
- **cancelAuction(): Permite cancelar la subasta antes de que existan ofertas.**
- **getBidCount(): Devuelve el número de ofertas realizadas.**
- **getBidHistory(uint offset, uint limit): Devuelve una página del historial de ofertas (paginación).**
- **getWinner(): Devuelve el ganador y el valor de la oferta ganadora.**

Seguridad y Buenas Prácticas

- **Uso de modificadores para controlar acceso y estado.**
- **Manejo seguro de transferencias de Ether siguiendo el patrón Checks-Effects-Interactions para evitar ataques de reentrancia.**
- **Control de errores y condiciones excepcionales con mensajes claros.**
- **Uso de eventos para notificar cambios de estado a los participantes.**
- **El historial de ofertas puede consultarse de forma paginada para evitar problemas de gas.**

Buenas Prácticas de Codificación

- **Variables y nombres de funciones en inglés: Facilita la comprensión y mantenimiento del código por parte de desarrolladores internacionales.**
- **Comentarios en español: Cada función y sección relevante del contrato incluye comentarios en español para que cualquier persona hispanohablante pueda entender fácilmente la lógica y el propósito del código.**
- **Mensajes de error descriptivos: Todos los require incluyen mensajes claros y específicos para facilitar la depuración y el uso correcto del contrato.**
- **Estructura modular y legible: El código está organizado en secciones lógicas y utiliza estructuras y modificadores para mejorar la claridad y seguridad.**
- **Evitar duplicidad de lógica: Se reutilizan estructuras y patrones seguros para evitar errores y vulnerabilidades.**

- **Uso de tipos y visibilidad adecuada: Las variables sensibles son privadas cuando corresponde y las públicas solo cuando es necesario para la transparencia.**
